DISSENY I DESENVOLUPAMENT D'UNA EINA DE SUPORT A LA PRESA DE DECISIÓ A LA REALITZACIÓ DE RESCATS DE MUNTANYA

Treball final de Màster

FIB, MEI

Director: Jaume Figueras
Ponent: Jordi Montero
Autor: Francesc de Paula de Puig Guixé

Agraïments

M’agradaria agrair a l’InLab FIB, i en concret al seu director Josep Casanovas, l’oportunitat d’haver desenvolupat aquest projecte i l’experiència i coneixements adquirits durant la meva estada que de bon segur em suposaran un gran impuls en el mon laboral

També m’agradaria agrair al meu director Jaume Figueras pel suport i seguiment al llarg del projecte i per introduir-me en el mon de les tecnologies GIS.

A en Jordi Montero professor ponent d’aquest projecte, agrair també el seu seguiment, suport i els seus coneixements. M’ha donat uns grans consells sobre com realitzar aquesta memòria.

Agrair a la Loreto Vaquero el seu suport durant la realització d’aquest projecte i sobretot en les ultimes hores de la redacció de la memòria

M’agradaria agrair el suport de tota la família, sobretot en aquest últim mes on no he passat per casa. Però també durant tota la durada del màster. Ajudant-me en tasques que són de la meva responsabilitat fent-me possible cursar aquest màster i realitzar el meu projecte

No puc acomiadar-me sense agrair el suport i dedicar aquest projecte al meu tiet, amic i company d’excursions Josep de Puig Viladrich que ens va deixar durant la realització d’aquest projecte.

Sense vosaltres aquest projecte no hauria estat possible.

Sincerament, gràcies

Abstract

Currently, the number of emergency mountain rescues in Catalonia is still very high, regardless of the preventive measures that have been taken during the last years. The support system for decision making during mountain rescue continues to depend solely on the knowledge of the GRAE team members, and an optimized computer-based support system still does not exist.

This project involves the design and development of a support tool for decision-making in mountain rescues (**ARM** Mountain Rescue Assistance). ARM is comprised of three pieces: (1) a proces that converts geographic mapping information into a graf format (2) a backend that optimizes rescue and emergency routes, and (3) an interface that interacts with the program user.

To that end we have developed scripts to validate the geographic information and convert the mapping data into a graf format. Then, the backend uses the graf information and Dijkstra's algorithm to optimize the emergency rescue routes. Finally, the interface collects the emergency information and gives it as an output to the user, the analysis results are in a format that can be easily understood by the emergency team.

This project takes into account the needs of the GRAE team, and has received invaluable support from the inLab FIB, Jaume Figueras and Jordi Montero. We believe that this social project will become a valuable tool in mountain rescues.

Resumen

En la actualidad, el número de rescates en montaña de Cataluña sigue siendo muy alto, independientemente de las medidas preventivas que se han tomado durante los últimos años. El sistema de soporte para la toma de decisiones durante los rescate de montaña continúa dependiendo únicamente del conocimiento de los miembros del equipo GRAE, y aún no existe un sistema de soporte optimizado automatizado.

Este proyecto tiene como objetivo el diseño y desarrollo de una herramienta de apoyo para la toma de decisiones en rescates de montaña (**ARM** Asistencia de Rescate en Montaña). ARM consta de tres partes: (1) un proceso que convierte la información de mapeo geográfico en un formato grafo (2) un *backend* que optimiza las rutas de rescate y emergencia, y (3) una interfaz que interactúa con el usuario del programa.

Con este fin, hemos desarrollado scripts para validar la información geográfica y convertir los datos de mapeo en un formato grafo. Luego, el *backend* utiliza la información de grafo y el algoritmo de Dijkstra para optimizar las rutas de rescate de emergencia. Finalmente, la interfaz recopila la información de emergencia y la entrega al usuario, los resultados del análisis están en un formato que el equipo de emergencia puede entender fácilmente.

Este proyecto tiene en cuenta las necesidades del equipo de GRAE y ha recibido un apoyo inestimable por parte del **InLab FIB**, Jaume Figueras y Jordi Montero. Creemos que este proyecto social se convertirá en una herramienta valiosa en los rescates de montaña.

Resum

En l'actualitat, el nombre de rescats a muntanya de Catalunya segueix sent molt alt, independentment de les mesures preventives que s'han pres durant els últims anys. El sistema de suport per a la presa de decisions durant els rescat de muntanya continua depenent únicament del coneixement dels membres de l'equip GRAE, i encara no hi ha un sistema de suport optimitzat automatitzat.

Aquest projecte té com a objectiu el disseny i desenvolupament d'una eina de suport per a la presa de decisions en rescats de muntanya (**ARM** assistència de rescat de muntanya). ARM consta de tres parts: (1) un procés que converteix la informació de mapeig geogràfic en un format graf (2) un *backend* que optimitza les rutes de rescat i emergència, i (3) una interfície que interactua amb l'usuari del programa.

Amb aquesta fi, hem desenvolupat scripts per validar la informació geogràfica i convertir les dades de mapeig en un format graf. Després, el *backend* utilitza la informació de graf i l'algoritme de Dijkstra per optimitzar les rutes de rescat d'emergència. Finalment, la interfície recopila la informació d'emergència i el lliurament a l'usuari, els resultats de l'anàlisi estan en un format que l'equip d'emergència pot entendre fàcilment.

Aquest projecte té en compte les necessitats de l'equip de GRAE i ha rebut un suport inestimable per part del **InLab FIB**, Jaume Figueras i Jordi Montero. Creiem que aquest projecte social es convertirà en una eina valuosa en els rescats de muntanya.

Índex

Índex d'il·lustracions

Índex de taules

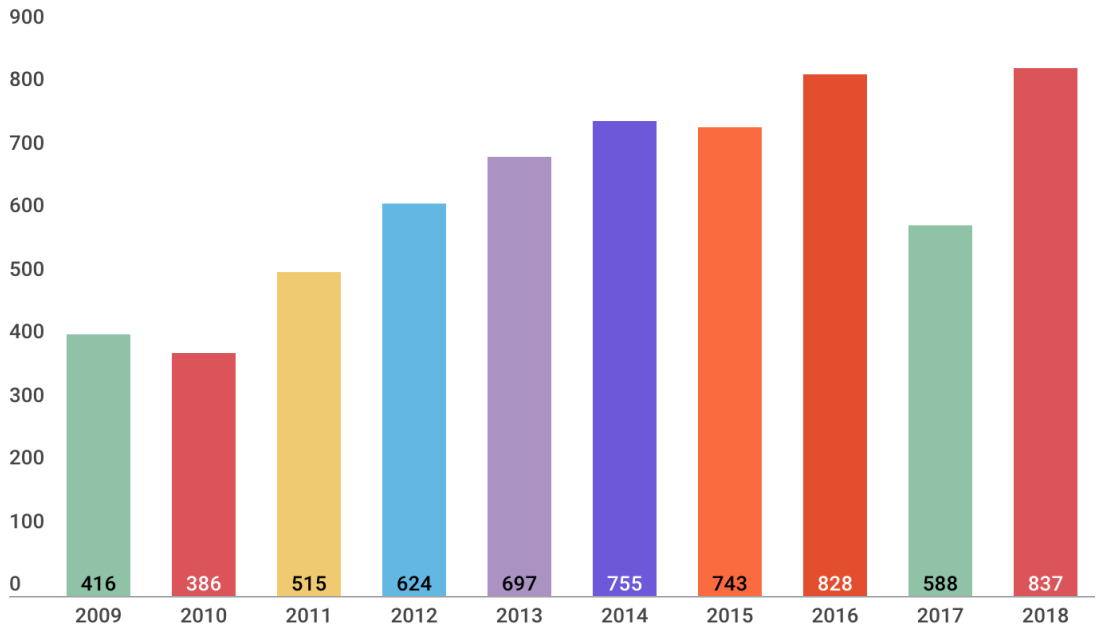
1 Introducció

1.1 Context

El **Grup d'Actuacions Especials** (**GRAE**) forma part del cos de Bombers de la Generalitat de Catalunya, estan especialitzats en salvaments i rescats en el medi natural i en llocs de difícil accés. El GRAE realitza cada any una mitjana de 600 rescats de muntanya sent el Parc Natural de Montserrat un dels espais naturals amb més sinistres. Aquestes intervencions suposen un cost econòmic important i la dedicació de recursos, escassos, que cal distribuir de manera molt curosa per cobrir tot el territori.

El gran nombre d'actuacions a Montserrat es deu al interès d'aquest massís tant emblemàtic, que fa venir gent de tota mena i indret. Des de turistes que no van preparats per pujar un cim, fins a escaladors experts que es troben amb algun imprevist.

Hi ha temporades on podem parlar de 2 o 3 rescats al dia a Montserrat. I ni el fet d'haver de pagar el rescat per negligència, ni els avisos de precaució sembla que redueixen aquesta xifra, tal i com es mostra en el següent graf.



Il·lustració 1: Salvaments de muntanya a Catalunya per anys[1]

[1] (3) Source: Interior, Salvaments de muntanya a Catalunya per anys

Fruit d'aquesta necessitat, s'han començat a implementar una sèrie de mesures per a millorar la seguretat de les persones que accedeixen al massís, amb actuacions que van des de facilitar l'orientació d'excursionistes perduts a millorar els protocols de rescat.

1.1.1 Montserrat, Avanços en els rescats

Els bombers de la Generalitat, la Federació d'Entitats Excursionistes de Catalunya (FEEC) i el Patronat de Montserrat van constituir un grup de treball per analitzar i millorar la seguretat dels excursionistes i escaladors que accedeixen al massís.

A part de mesures tan informatives, divulgatives, com d'adequació i formació també s'han realitzat les següents actuacions:

1.1.1.1 Punts de cobertura del telèfon d'emergències del 112

El departament d'Interior i el Patronat de la muntanya de Montserrat van signar un conveni de col·laboració en el qual es preveia la instal·lació a la muntanya de Montserrat d'una cinquantena de senyals informatius per indicar els punts de cobertura del telèfon d'emergències 112. D'aquesta manera, visitants i excursionistes saben quines són les zones on hi ha possibilitat d'activar als serveis d'emergència, en cas de necessitat.

1.1.1.2 Elaboració d'una Cartografia Operativa d'Emergències (COE)

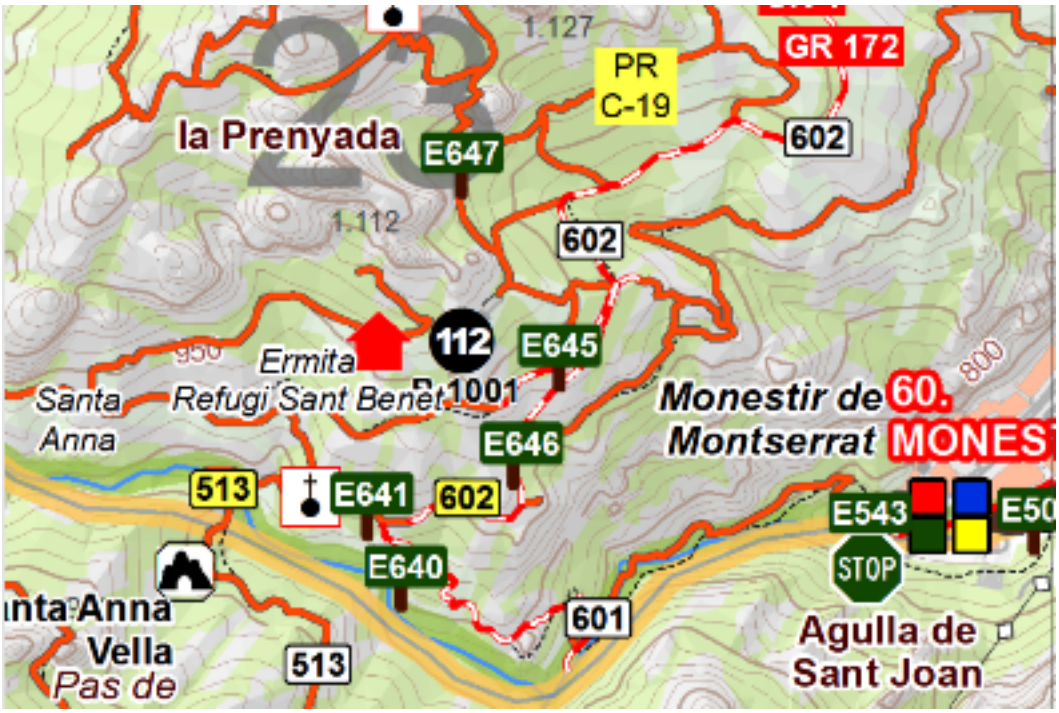
Per les seves característiques orogràfiques, meteo-climàtiques i socials, Montserrat és un punt singular en l'àmbit de les emergències sent necessari l'elaboració d'una eina cartogràfica específica per a la muntanya de Montserrat. Aquesta cartografia ha estat realitzada pel cos de Bombers de la Generalitat de Catalunya, localitzant tots els punts considerats d'interès per part dels serveis d'emergències.

Aquesta nova eina cartogràfica inclou, a part de la informació de tipus exclusivament geogràfic, aquells elements que són importants a l'hora de definir la resposta operativa com poden ser: fonts d'aigua, passos protegits amb cadenes o punts de reagrupament.

Val a dir que aquesta documentació geogràfica no està tancada i requerirà de diferents actualitzacions que permetin donar una millor resposta atenent els canvis que esdevenen dins del parc. En aquest sentit els bombers apel·len a la ciutadania a què col·labori per a ampliar aquesta informació, esdevenint una eina col·laborativa.

La COE de la muntanya de Montserrat es pot trobar en els formats (PDF, mapa d'agulles, KMZ, *shapefile*):

Aquests recursos estan disponibles pels equips de coordinació de rescat, així dons s'integraran amb els programaris actuals que utilitzen els equips de salvament, i per tant s'haurà de considerar en l'elaboració d'aquest treball i degudament analitzat.



Il·lustració 2: retall de COE de Montserrat[2]

1.1.2 Sistema actual de suport a la presa de decisions

El sistema actual, basa l'enrutament en una heurística A* i té una sèrie de mancances tecnològiques que han donat lloc a la realització d'aquest TFM. Els principals problemes reportats pel cos de bombers són:

- Rendiment de l'aplicació
- Precisió de l'aplicació
- No és una solució intermodal
- Estructura de dades
- No disposa d'una interfície gràfica

[2] (4) Source: Interior Cartografia operativa d'emergències de Montserrat

Previ al desenvolupament del treball s'ha realitzat un estudi detallat del sistema actual per tal de comprovar el nivell de severitat dels problemes esmentats anteriorment.

Pel que fa al rendiment de l'aplicació**,** es va dissenyar un conjunt d'escenaris d'experimentació que permetessin mesurar els indicadors de temps d'execució i temps invertit per node en funció de l'espai de cerca. Els resultats obtinguts es poden observar en la següent taula.

Mecanisme de cerca	**Mida del graf**	**Nombre d'arcs**	**Cost ruta**	**Vehicle**	**Temps d'execució (s)**	**Nodes visitats**	**Temps per node (ms)**
Cost Uniforme	889	1010	1:38:34	Peu	1.8	266	52.26
A*	889	1010	1:38:34	Peu	1.5148	199	55.52
Cost Uniforme	3007	3771	1:41:46	Peu	93.21	1843	478.73
A*	3007	3771	1:41:46	Peu	28.16	843	274.97
Cost Uniforme	3007	3771	2:15:25	Peu	69.83	1520	426.64
A*	3007	3771	2:15:25	Peu	32.47	884	310.97
Cost Uniforme	3007	3771	0:43:20	4x4 Peu	147.7	2463	579.46
A*	3007	3771	0:43:20	4x4 Peu	110.75	2136	495.18

Taula 1: Comparativa dels temps d'execució del sistema actual en diferents condicions

D'on es pot concloure que el sistema actual pot trigar més de dos minuts en retornar el camí mínim per arribar el punt de rescat, evidentment aquest temps es pot acceptar per a solucions d'un sol vehicle, **solució no intermodal**.

Ara bé, en un entorn escalable on es vulgui considerar tots els vehicles de transport terrestre dels bombers, indiferentment de la seva posició, aquests dos minuts poden ser un greu coll d'ampolla que caldrà solucionar en el nou sistema..

Pel que fa a les **estructures de dades** es va realitzar una comprovació de les mateixes i el grau de connectivitat dels grafs definits en ella, detectant-se certes inconnexions: 85 vèrtexs de 3007 es trobaven desconnectats de la resta.

Il·lustració 3: Exemple de vèrtexs no adjacents visualitzat en JOSM

Finalment esmentar que el sistema actual funciona mitjançant línia de comandes, amb els problemes afegits que això pot comportar a un usuari no especialitzat en aquest tipus d'interfície i que la solució que s'obté no és multimodal (sent aquest un requisit directe dels bombers).

De tot aquest anàlisi és pot concloure que el sistema actual no es una bona eina pel suport a la presa de decisions.

1.2 Justificació

El cos de bombers ja té una cartografia, sobre la qual planificar les actuacions a Montserrat, i sistemes amb els quals obtenir la coordenada exacte dels accidentats. Però actualment, la decisió de quina ruta es segueix encara es pren majoritàriament en base a l'experiència dels bombers, el que no sempre garanteix que el camí decidit sigui l'idoni.

Tal com s'ha esmentat en la secció anterior, el sistema actual és lent, no disposa de una interfície d'usuari sent necessari l'execució des de línia de comandes, el que no es pràctic pels bombers, i està focalitzat en el parc Natural de Montserrat.

El darrer hàndicap que justifica aquest treball final de màster es la impossibilitat d'escollir quin dels diferents equips o recursos del cos de bombers de la Generalitat de Catalunya es troba més ben situat per a realitzar el rescat.

Així doncs, partint de la cartografia de bombers i del coneixement del funcionament i mancances del sistema actual com a punt de partida d'aquest projecte. Realitzarem un **sistema de suport a la presa de decisions en rescats de muntanya** que sigui pràctic pels bombers i que resolgui totes les mancances del sistema actual. Aquest nou sistema de suport a la presa de decisions l'anomenarem **Assistència al Rescat de Muntanya** (**ARM**).

1.3 Definició del problema

El cos de bombers necessita una eina de suport per trobar la ruta òptima considerant factors com el tipus de via, desnivell del terreny, vehicles utilitzats i els camins de retorn en el parc de Montserrat.

Actualment, la decisió de quina ruta es segueix es pren mitjançant únicament l'experiència dels bombers, el que no sempre garantitza que el camí decidit sigui l'idoni o que hi hagi una excessiva dependència dels bombers més experimentats.

Per la majoria de rescats, en l'actualitat i en el cas que ens han plantejat, els bombers ja tenen la coordenada exacta on s'ha produït la incidència via geolocalització mòbil o per la matrícula dels punts de cobertura del telèfon d'emergències del 112.

Precisament, una de les eines que els ha resultat molt útil en la realització de tasques de rescats, és el mòbil. Emprant eines com OruxMaps[3] poden consultar la seva cartografia, o la disposició dels vehicles a la zona.

Per altra banda, la cartografia dels bombers és molt completa, defineix si els camins son aptes per camions, 4x4 o només són transitables a peu. Aquesta cartografia, també compta amb la

[3] Source: Santi Lleonard, , cap de la unitat territorial II de la Regió d'Emergències Sud

senyalització de fonts, camins tancats al públic, punts de reunió... que estan degudament representats.

La posició dels vehicles i equips, la comparteixen a través d'un XML sincronitzat mitjançant de *DropBox*[4]. Ens podem trobar amb més d'un possible punt de sortida en funció de la posició de les diverses unitats del cos. Aquestes unitats diferents comptaran també amb diversos vehicles, de tal manera que podem trobar equips situats a punts diferents i on cadascun pot comptar amb vehicles de diferent o del mateix tipus incloent com tipus un desplaçament a peu. El treball que ens proposem realitzar serà molt útil per germanitzar el resultat més idoni i racionalitzar els recursos i la seguretat en el rescat.

1.4 Abast

Aquest TFM té per objectiu el disseny i desenvolupament d'una eina de suport a la presa de decisions quan es produeix una operació de rescat.

La solució proposada optimitzarà i, conseqüentment, permetrà una millor gestió dels diferents equips que poden participar en una operació de rescat de muntanya a excepció dels helicòpters que quedaran fora de l'estudi. És a dir s'estudiarà vehicles tot terreny, vehicles mèdics i camions de bombers. Contemplarà el problema de la darrera milla, en el sentit de que els punts de trobada poden ser vistos com els "*hubs*" de els serveis de logística, i donem les possibles alternatives des del *hub* fins a la intervenció final dels bombers.

L'aplicació final permetrà calcular les diferents alternatives de rescat, rutes, així com la seva visualització sobre un mapa de dos dimensions per tal que els equips de coordinació tinguin una idea de les millors solucions per cost (en temps o dificultat).

El nou sistema tindrà en compte: les restriccions d'accés i limitacions de velocitat en funció del tipus de vehicle i tipus de via, i la velocitat de desplaçament a peu tenint en compte el desnivell del terreny.

L'eina ha de permetre fer això sense requerir que l'usuari tingui coneixements específics d'informàtica o programació. Donant una visualització intuïtiva i ràpida per tal que sigui útil en situacions d'emergència

[4] Source: Santi Lleonard, , cap de la unitat territorial II de la Regió d'Emergències Sud

L'elaboració de la cartografia necessària sobre la qual treballarà l'algorisme no forma part del present projecte, ja que aquesta serà subministrada pel cos de bombers. No obstant, es realitzarà un tractament d'aquestes dades per tal de resoldre les mancances detectades en l'estudi previ del sistema actual i establir un format que s'adeqüi al nou algoritme.

La solució final desenvolupada serà escalable per tal de donar cobertura a d'altres zones de baixa, mitja o alta muntanya e integrable amb les aplicacions actuals que fan servir els equips de salvament.

1.5 Objectius

A continuació es descriuen els objectius que s'esperen assolir en aquest treball:

1. El sistema donarà diverses solucions optimitzades pel rescat de muntanya a Montserrat pel cos de bombers
2. El sistema tindrà en compte desplaçaments en camió, amb 4x4 i a peu.
3. El sistema serà escalable a tot Catalunya
4. El sistema permetrà explorar l'arbre de possibilitats amb el mínim cost en temps d'accés, recursos i desnivell acumulat a peu
5. El sistema mostrarà de forma clara e intuïtiva els resultats
6. El sistema donarà eines per facilitar la decisió entre futures rutes
7. El sistema ha de ser escalable en un futur pròxim a nivell de funcionalitats com afegir helicòpters com a vehicles, el *geofencing* d'àrees i una possible integració amb QGIS

1.6 L'equip

Aquest projecte es realitza per personal del laboratori d'Innovació de la Facultat d'Informàtica de Barcelona de la UPC (Inlab FIB) en col·laboració amb membres del cos de bombers

1.6.1 inLab FIB

L'inLab FIB UPC és el laboratori d'innovació i recerca de la Facultat d'Informàtica de Barcelona de la UPC, especialitzat en aplicacions i serveis basats en les últimes tecnologies, en més de 40 anys d'experiència col·laborant en projectes innovadors, desenvolupant solucions a mida per a entitats públiques i privades, i oferint el servei de laboratoris d'aprenentatge especialitzats en l'enginyeria informàtica.

Els membres de l'inLab FIB que participen en aquest projecte són:

1.6.1.1

- Jaume Figueras, cap de projectes GIS, és graduat en Informàtica l'any 1998. Duu a terme la seva investigació en control automàtic i en simulació i optimització per ordinador. Participa en diferents projectes industrials, com l'optimització de consum energètic de les línies de tramvia a Barcelona conjuntament amb TRAM i SIEMENS i el desenvolupament de *tooPath* un sistema web gratuït que permet el seguiment de dispositius mòbils. És també el representant local d'OSM a Catalunya i participa en diferents projectes *FOSS*
- Jordi Montero, ponent d'aquest projecte és enginyer en informàtica per la FIB. Ha treballat en diferents projectes multidisciplinaris de modelització, simulació i optimització per a diferents empreses i àmbits de la logística, producció i serveis en col·laboració amb els doctors Casanovas, Guasch i Fonseca, entre d'altres professors de la UPC, UB i UAB.
- Francesc de Puig, estudiant del MEI de la FIB i graduat el 2017 en enginyeria informàtica per la FIB.

1.6.2 Cos de bombers

Aquest projecte està codirigit i assessorat per:

- Santi Lleonard, cap de la unitat territorial II de la Regió d'Emergències Sud i Codirector d'aquest projecte.
- David Colomé Sangrà, cap de Parc dels Bombers Voluntaris de Collbató de la Regió d'Emergències Metropolitana Sud.

2 Metodologia

Gairebé per tots els projectes que es desenvolupen a l'inLab FIB es fa servir una metodologia àgil. Generalment una metodologia basada en Scrum. Per tant la proposta d'aquest projecte és seguir la mateixa metodologia adaptant-la a les nostres necessitats.

2.1 Àgil

Les metodologies àgils o processos àgils de desenvolupament de programari (com, per exemple, XP, Scrum, DSDM, Cristal, etc.) són aquelles metodologies de desenvolupament que es basen en l'adaptabilitat de qualsevol canvi com a mitjà per augmentar les possibilitats d'èxit d'un projecte.

Es diu que àgil és més aviat una manera de pensar que una metodologia. Alistair Cockburn va suggerir que una metodologia és el conjunt de convencions que un equip accepta de seguir[5]. Això significa que cada equip tindrà la seva pròpia metodologia. Així, les metodologies àgils són les convencions que un equip escull seguir de manera que segueixi els **valors** i **principis** àgils.

Els **valors** d'àgil queden plasmats en el manifest àgil on consta que es promou:

- Els individus i les seves interaccions per sobre dels processos i les eines
- El programari que funciona per sobre de la documentació exhaustiva
- La col·laboració amb el client per sobre de la negociació de contractes
- La resposta davant del canvi per sobre de seguir un pla tancat

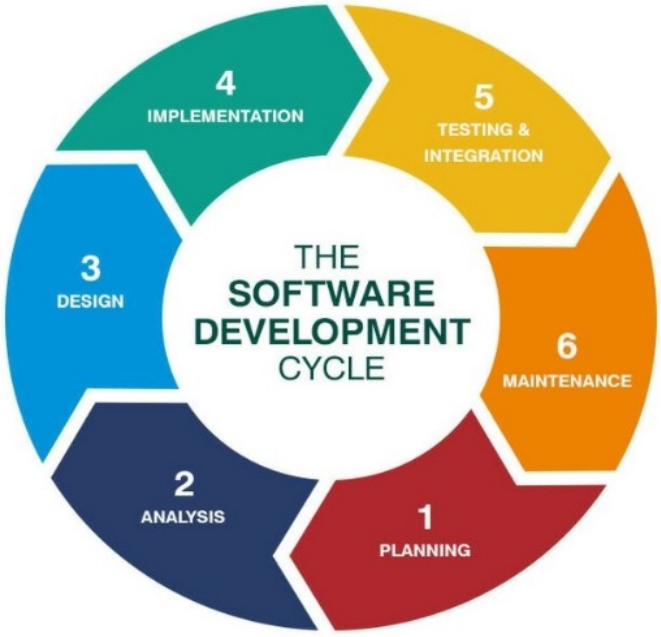
Per seguir això es basen en 12 **principis**:

1. La nostra principal prioritat és satisfer al client mitjançant el lliurament primerenc i continu de programari que aporti valor.
2. Acceptem de bon grat canvis als requisits, inclús si arriben cap el final del desenvolupament. Els processos àgils aprofiten el canvi per a donar un avantatge competitiu al client.
3. Lliurem amb freqüència programari que funcioni, des d'un parell de setmanes fins a un parell de mesos, amb preferència per l'escala de temps més curta.
4. La gent de negoci i els desenvolupadors han de treballar junts de manera quotidiana durant tot el projecte.

[5] (9)Source: Agilealience

5. Construïm projectes amb l'ajuda d'individus motivats. Els donem l'entorn i el recolzament que necessiten i confiem en ells per fer la feina.
6. El mètode més eficient i efectiu de comunicar informació cap a i dins d'un equip de desenvolupament és la conversa cara a cara.
7. El programari que funciona és la principal mesura de progrés.
8. Els processos àgils promouen el desenvolupament sostingut. Els promotors, desenvolupadors i usuaris han de ser capaços de mantenir un ritme constant de manera indefinida.
9. L'atenció contínua a l'excel·lència tècnica i al bon disseny millora l'agilitat.
10. La simplicitat, l'art de maximitzar la quantitat de feina que no es fa, és essencial.
11. Les millors arquitectures, requisits i dissenys emergeixen d'equips auto organitzats.
12. En intervals regulars, l'equip reflexiona sobre com ésser més efectiu, s'afina i ajusta el seu comportament d'acord amb això.

Per tal de seguir les metodologies àgils solen funcionar en cicles de periodicitat regular:

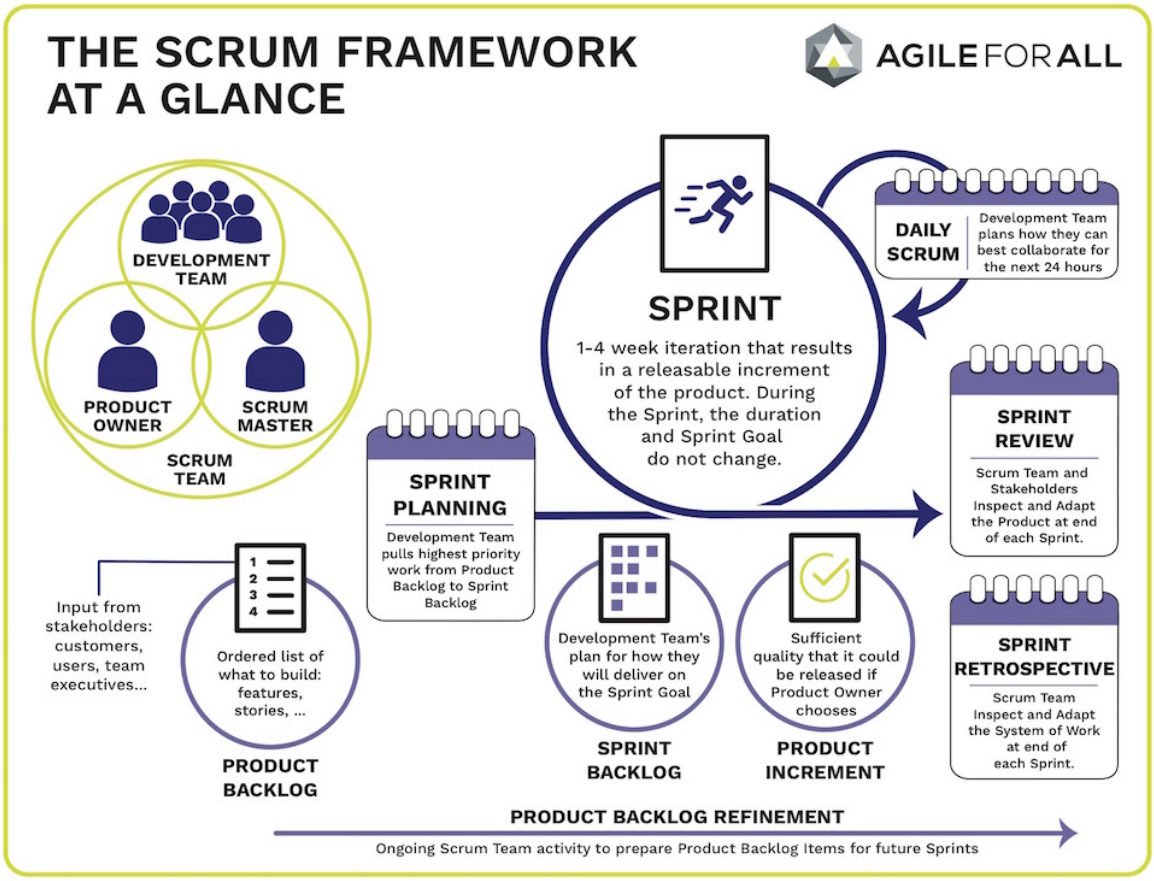


Taula 2: Cicle Agil del desenvolopament del software[6]

2.2 Scrum

Scrum segons els seus creadors és un *framework* en el qual les persones poden abordar problemes d'adaptació complexos, alhora que ofereixen de manera productiva i creativa productes de major valor possible. Per tant no es basa en un conjunt de regles que s'han de complir, si no que es pot adaptar a les necessitats de l'equip

[6] (13)Source: Agilealience, the software development cicle



Il·lustració 4: Visió general de la metodologia Scrum[7]

Scrum es basa en la teoria del control de processos empíric o empirisme. L'empirisme afirma que el coneixement prové de l'experiència i pren decisions basades en el que es coneix. Scrum utilitza un enfocament iteratiu i incremental per reduir el risc i disposar dels canvis associats a necessitats detectades pel client en temps de desenvolupament.

En Scrum s'assignen rols per tal d'aconseguir que els equips siguin capaços d'autoorganitzar-se i tinguin un coneixement transversal de tot el projecte. Els rols definits a la Guia Scrum són:

- *Product Owner*: Jaume Figueras és responsable de maximitzar el valor del producte resultant del treball de l'equip de desenvolupament.
- *Development team*: Francesc de Puig fa la feina d'oferir un increment potencial del producte "Fet" al final de cada sprint, de manera autònoma en quan a l'organització i mètode

[7] (15) Source: Agilealinece, Scrum

- *Scrum Master*: Jordi Montero és responsable de promoure i donar suport a Scrum tal com es defineix a la Guia Scrum. Els Scrum Masters ho fan de manera que tothom entengui la teoria, les pràctiques, les regles i els valors de Scrum

Les iteracions a Scrum es centren en el els Sprints, generalment a l'inLab FIB treballem en *sprints* de dos o tres setmanes, però en aquest projecte les hem fet durar una setmana per potenciar la comunicació.

Dintre les adaptacions s'han realitzat els *dailys* de forma online, amb la participació del Scrum Master ja que l'equip de desenvolupament només estava format per una persona.

Com els sprints són d'una setmana, cada setmana, s'ha presentat el progrés al director i al ponent del projecte. L'esdeveniment de Scrum conegut com a *Sprint Revew*. La forma de presentar el progrés han estat en forma de presentació o demostració, segons ens facilites validar el avenç realitzat.

Les fases de *Sprint planning* i *Sprint Retrospective* s'han fet tenint en compte les característiques de l'equip i d'aquest projecte.

El projecte arranca amb una reunió de *kick-off* amb els responsables per part del cos de bombers que ens expressen els seus neguits i desitjos per a la millora del sistema actual, establint aquest punt d'inici com el punt de recollida de requisits funcionals i no funcionals del sistema que es desenvoluparà.

A mesura que s'han anat superant els aspectes tècnics i quan el projecte arriba a una fase on el més rellevant és la vessant pràctica del mateix, s'han concertat reunions amb els caps dels bombers per tal d'obtenir el seu feedback.

El projecte s'ha desenvolupat en 21 setmanes, és a dir 21 sprints d'una setmana. Del 4 de Febrer fins el 21 de Juny

2.3 Eines de gestió

Pel desenvolupament d'aquest projecte s'han treballat en diferents eines de gestió que han permés garantir el control de versions i el seguiment de les tasques.

2.3.1 Github

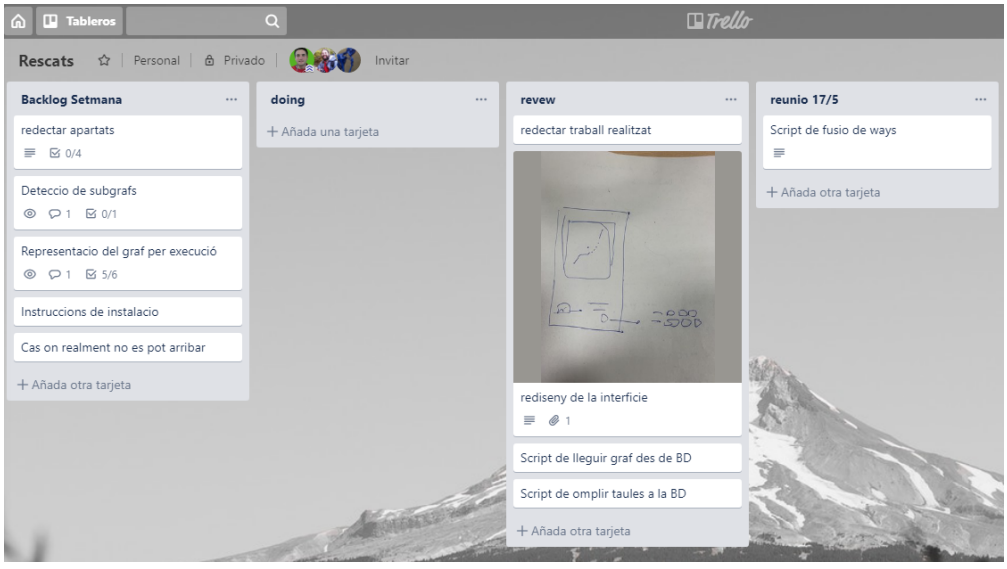
Tot i ser un únic desenvolupador, s'ha optat per treballar amb Github com a sistema de control de versions distribuït, que ens permetia tenir un control de les versions encara que no ha estat necessari disposar de totes les seves funcionalitats, com ara: *branches, merge o pull request.*

2.3.2 Trello

Per gestionar a nivell d'equip el procés de desenvolupament s'ha fet servir Trello, seguint el flux de treball habitual en aquest tipus de projectes.

BACKLOG → TO DO → DOING → REVIEW → DONE

Per cada Sprint un cop tenim estimades i decidides les tasques a realitzar, extretes del BACKLOG, a la llista de TODO, les anàvem passant a DOING o REVIEW segons avançàvem, a cada sprint fins assolit l'estat desitjat de DONE que ens permetia verificar el treball fet.



Il·lustració 5: Captura de Trello durant el progres del projecte

3 Estat de l'art

Com es menciona en apartats anteriors, un dels punts del nou sistema es que ha de ser capaç de combinar diferents tipus de vehicles per tal d'arribar al punt de rescat, així doncs hem d'estudiar solucions de transport multimodal.

3.1 El transport multimodal

El transport multimodal és aquell en què cal emprar més d'un tipus de vehicle per transportar la mercaderia des del seu lloc d'origen fins a la destinació final.

D'acord amb el concepte general de transport multimodal, és possible transportar càrrega per mitjans multimodals a granel, amb o sense contenidors o efectuar operacions de transport multimodal domèstic. Dins d'aquest marc global, distingim:

- **transport intermodal**: utilitzant diversos tipus de transport però utilitzant una única mesura de càrrega
- **transport combinat**: diferents mitjans dins d'una mateixa cadena de transports.

Com dintre del marc d'aquest treball no entrarem a valorar les capacitats de càrrega el nostre algoritme serà del tipus transport intermodal

3.1.1 Transport intermodal

La Comissió Europea defineix la intermodalitat com «la característica del sistema de transport que permet utilitzar almenys dues maneres de forma integrada en la cadena de transport 'porta a porta'»

En l'àmbit del transport de mercaderies, es denomina transport intermodal a l'articulació entre diferents modes de transport utilitzant una única unitat de càrrega, generalment contenidors, per tal de realitzar més ràpida i eficaçment les operacions de transbord de materials i mercaderies, durant el trasllat de la càrrega des d'un punt d'origen fins a un punt de destinació.

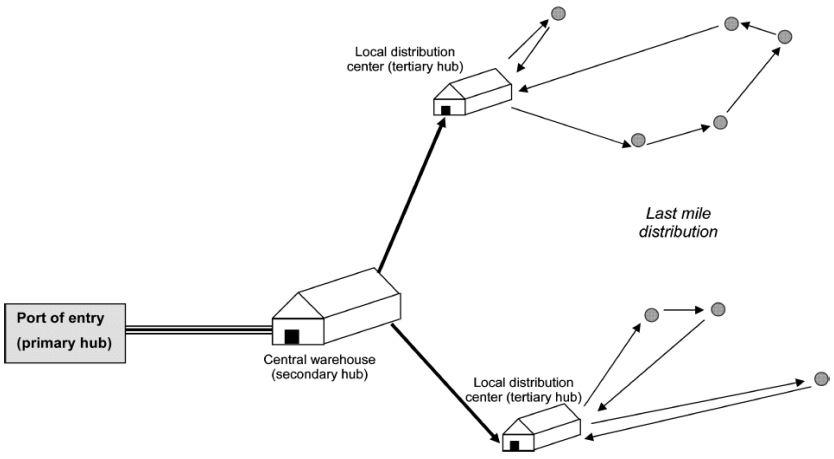
Tradicionalment, el moviment de mercaderies es considera com una sèrie de viatges independents (per carretera, ferrocarril, marítim o aeri), que poden creuar-se en algun punt on la mercaderia es transvasa d'un mode de transport a un altre. Però avui en dia, el transport de mercaderies es considera un flux constant entre un origen i un destí (transport 'porta a porta'), amb un concepte integrat de transport, basat en la connexió dels nodes i xarxes de transport.

Dins la tipologia del transport intermodal, ens interessa particularment el problema de la darrera milla, que defineix la gestió de la cadena de subministrament i la planificació del transport per descriure el moviment de persones i béns des d'un centre de transport fins a un destí final a la llar.

Per una banda ens fixem en el problema de l'ultima milla per la tipologia del desplaçament. Generalment en el transport intermodal parlem de vaixells o trens de carrega, però en l'ultima milla la mercaderia ha d'arribar al destí final. Per aquesta fi l'escala dels vehicles i les vies a emprar són mes semblants a les de la tipologia de rescat, on disposem de camió, 4x4 i desplaçament a peu.

També ens em de fixar en el problema de l'ultima milla el sentit de que els punts de trobada poden ser vistos com els "hubs" de els serveis de logística, i donem les possibles alternatives des del hub fins a la intervenció final dels bombers

De fet en quant a la infraestructura es refereix el problema de la ultima milla s'ha estudiat en el transport d'ajudes humanitàries. Els subministraments d'ajuda de vegades poden arribar a un centre de transport central en una zona afectada, però no es poden distribuir per danys causats per un desastre natural o per la manca d'infraestructura. Així doncs no es pot comptar amb trens o grans infraestructures, si no més aviat en carreteres i camins com al nostre problema.



Il·lustració 6: Esquema de una cadena d'ajudes humanitàries[8]

[8] (19) Source: UNDP Disaster Management Training Programme, Logistics Module (1st ed.), p.18

3.2 Solucions existents

Previ el desenvolupament i com a part de l'estat de l'art es realitza un estudi per avaluar si la solució que proposem podria adaptar-se a algun sistema actual, doncs com ja hem vist en l'estudi previ el sistema actual no es pot considerar un bon DSS (*Decision Support System*)

3.2.1 Solucions estudiades d'interès

3.2.1.1 OpenTripPlaner:

OpenTripPlaner és un planificador de viatges multimodal *open-source* que fa servir dos algoritmes diferents: A* i Jerarquies de contracció. Originalment, OpenTripPlanner feia servir només l'algoritme de Dijkstra's i A* amb distàncies Euclidianes per enrutar viatges en tots els medis de transport. Però el rendiment era molt lent en grafs grans.

Tot i que ha estat interessant estudiar OpenTripPlanner a nivell de com solucionar el problema l'enrutament multimodal, no és pràctic en el sentit de que no te en compte la diferència d'alçada en les distàncies. Que en un rescat de muntanya causa un efecte important.

Tampoc és un DSS ja que no esta pensat per donar diverses solucions al problema per que l'usuari triï sinó l'òptima.

3.2.1.2 OptaPlaner

OptaPlanner es un software escrit en JAVA que fa servir la cartografia del model OSM. Tracta el problema del viatjant de comerç (Travelling Salesman Problem) , el problema de l'enrutament de vehicles, entre altres.

Cobreix qualsevol tipus de programació de la flota, com ara l'encaminament d'avions, camions, autobusos, taxis, bicicletes i vaixells, independentment de si els vehicles transporten productes o passatgers o si els conductors estan prestant serveis.

Aquest cas, no s'adapta el nostre problema, tot i que es un DSS, està molt enfocat en flotes comercials i com l'anterior no té en compte l'altura

3.2.1.3 Citymapper

Citymapper és una aplicació mòbil de *web mapping* i transport públic. Integra informació de tots els mitjans de transport urbà, especialment transport públic; tot i que també afegeix opcions per anar a peu, en bici, o en sistemes de transport compartit (bicicletes, motos, cotxes o fins i tot patinets). És capaç de donar opcions combinades Les dades s'obtenen per contribucions dels usuaris, de dades obertes, o per personal local.

Pel nostre cas, essent tant enfocada al transport urbà ens és difícil adaptar-la al nostre sistema i de nou no utilitza l'altura.

3.2.1.4 PyrouteLib

Llibreria de *Python* del programa d'enrutament pyroute, no es multimodal ni és un DSS. S'ha estudiat en el marc d'aprofitar aquesta llibreria per càlculs que no requereixin de la multimodalitat ni d'altures, com l'enrutament dels vehicles fins els punts de trobada a Montserrat.

Finalment el seu baix rendiment ens va fer descartar-la. Es va fer servir en l'estudi de solucions de *Python*, fent-nos veure que una solució integra en *Python* no era viable

3.2.1.5 OSMGraph

Llibreria de *Python* d'exploració de rutes sobre cartografia OSM. Igual que PyrouteLib, no es multimodal ni és un DSS i s'ha estudiat en el marc d'aprofitar aquesta llibreria per càlculs que no requereixin de la multimodalitat ni d'altures.

També com PyrouteLib el seu baix rendiment ens va fer descartar-la i es va fer servir en l'estudi de solucions de *Python* on vam veure que les solucions íntegrament en *Python* no eren prou eficients.

3.2.1.6 Brouter

BRouter és un sistema d'enrutament dissenyat per calcular rutes de ciclisme òptimes. Fa servir el model de OSM i dades d'elevació. Està disponible com a servei web i aplicació Android. L'aplicació per a Android és compatible amb OsmAnd, OruxMaps i Locus Map com a servei d'enrutament seleccionable. La generació de rutes es fa fora de línia.

Les conclusions a les que podem arribar amb aquest anàlisi és que tot i que compleix el requisit de emprar dades d'elavació. Esta molt enfocat al ciclisme i no a solucions multimodals i a

diferencia dels dos anteriors, com no està enfocat a altres vehicles tampoc ens serveix per fer el Routing dels vehicles fins el parc..

3.2.2 Conclusions de les solucions estudiades d'interès

Veient que qualsevol de les solucions exposades requeriria una feina d'importació de les dades emmagatzemades del COE, conjuntament amb l'exposat anteriorment reforça la idea de cara la realització d'aquest projecte la millor opció és una solució *adhoc* que integri una UI específica i algoritmes a mida, tant pel que fa al tractament de les dades del COE com a l'enrutament intermodal.

3.3 Algoritmes de càlcul de rutes

Havent explorat les solucions existents hem vist que ens cal explorar algoritmes de càlcul de rutes, enfocats en el problema del camí més curt. Els algoritmes més importants per resoldre aquest problema són els recollits a la següent taula.

Nom	Funció	Aplica
Algorisme de Dijkstra	Resol el problema dels camins més curts entre dos vèrtexs, des d'un origen i un únic destí	Si, podem trobar com anar a un punt des de diversos orígens
Algorisme de Bellman-Ford	Resol el problema dels camins més curts des d'un origen si la ponderació de les arestes és negativa.	Si, però és menys eficient que dijkstra donat que no tenim pesos negatius
Algorisme de Cerca A*	Resol el problema dels camins més curts entre un parell de vèrtexs utilitzant l'heurística per intentar agilitzar la cerca	Si, però hem de tenir en compte que es depenent de una heurística
Algorisme de Floyd-Warshall	Resol el problema dels camins més curts entre tots els vèrtexs.	No ja que no hem de passar per tots els punts
Algorisme de Johnson	Resol el problema dels camins més curts entre tots els vèrtexs i pot ser més ràpid que el de Floyd-Warshall en grafs de baixa densitat	No ja que no hem de passar per tots els punts
Algoritme de Viterbi	Tracta de buscar la seqüència d'estats(nodes) més probable.	No ja que no tractem quina és la seqüència de nodes més probables
Contraction Hierarchies	Resol el problema dels camins més curts fent servir una tècnica per accelerar l'encaminament de camins més curts mitjançant la primera creació de versions "contractades" precomputades del gràfic de connexió	Si, però em de tenir en compte que el preprocés pot descartar arestes que no podem estar segurs que no necessitarem

Taula 3: Anàlisi dels algoritmes de càlcul de rutes

Per les característiques d'aquest treball ens interessa estudiar **Dijkstra**, **A*** i **Contraction Hierarchies**.

3.3.1 Algoritmes estudiats

3.3.1.1 Dijkstra

L'algoritme de Dijkstra, conegut també com a algoritme de camins mínims, és un algorisme per la determinació del camí més curt donat un vèrtex origen a la resta de vèrtexs en un graf dirigit i amb pesos a cada aresta..

El procés que segueix aquest algorisme és anar explorant tots els camins més curts que parteixen del vèrtex origen i que porten a tots els altres vèrtexs, quan s'obté el camí més curt des del vèrtex origen, a la resta de vèrtexs que formen el graf, l'algorisme s'atura. No funciona en grafs amb arestes de cost negatiu

L'ordre de complexitat de l'algorisme és: O $(|V|^2+|E|) = O(|V|^2)$ sense utilitzar cua de prioritat, $O((|L|+|V|) \log|V|)$, utilitzant cua de prioritat[9]

També hi ha una variant per exploracions punt a punt anomenat **Dijkstra bidireccional** on s'explora el graf tant des del punt d'origen com des de el punt destí i s'acaba l'execució quan els dos es creuen, el camí que uneix els dos dijkstres passant pel punt de creuament es el camí mínim, i cap altre camí serà millor.

3.3.1.2 Algoritme A*

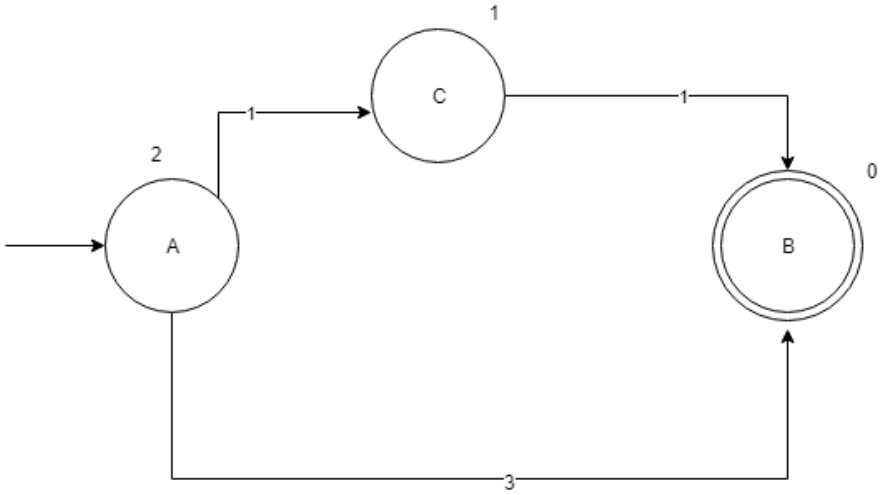
A * és un algorisme que utilitza una heurística per a solucionar el problema de la traçabilitat de trajectòries i en el recorregut de grafs, que és el procés de trobar un camí entre diversos punts, anomenats nodes.

Gaudeix d'un ús generalitzat a causa del seu rendiment i precisió. No obstant això, en els sistemes pràctics d'encaminament de viatges, generalment s'ha sobrepassat per algorismes que poden pre-processar el gràfic per aconseguir un millor rendiment.

El procés que segueix aquest algorisme és anar explorant tots els camins que parteixen del vèrtex origen i que porten al node que més ens acosta a l'objectiu segons l'heurística que hem aplicat. S'atura quan arribem al node objectiu.

A diferencia de Dijkstra no soluciona el problema del camí mínim, dons es pot donar el cas que doni un camí que tot i ser correcte no compleix el requisit de ser mínim. Com es mostra en el cas següent on A és l'origen B el destí i A* seleccionaria l'aresta amb cost 3 ja que l'heurístic dóna un valor més petit 0 al node B per ser el destí que el node C amb valor 1 quan el camí mínim seria passar pel node C i arribar a B amb cost 2.

[9] (48)Source: Shortest Path, Mehlhorn, Kurt; Sanders, Peter



Il·lustració 7: Exemple de càlcul amb heuristica

La complexitat computacional de l'algorisme està íntimament relacionada amb la qualitat de l'heurística que s'utilitza per resoldre el problema. Si es fa servir una heurística de mala qualitat el cost pot ser exponencial. En el millor cas possible l'algoritme s'executarà en temps lineal.

3.3.1.3 Contraction Hierarchies

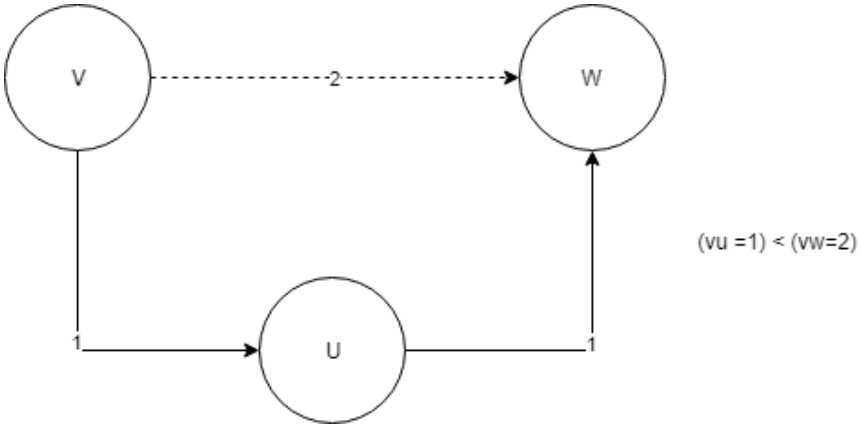
Publicat el 2008, actualment és un dels algorismes per la determinació del camí més curt coneguts més eficients en temps de consulta.

Per jerarquies de contracció s'ha de fer un preprocés, en el que es dona un ordre als nodes del graf.

Seguidament iterant cada node d'ordre inferior a superior es seleccionen els seus nodes adjacents d'ordre superior i es troba si el camí més curt entre cada parell passa pel node actual

Si es compleix que el camí més curt entre cada parell passa pel node actual, s'afegirà una drecera. La drecera serà el camí entre els dos nodes adjacents amb cost valor del camí mínim que els uneix.

En acavar el preprocés s'elimina el nodes exclosos en la drecera. De no fer això l'exploració del graf en Dijkstra mai agafaria la drecera dons els costs individuals de les arestes que componen la drecera son inferiors al cost de la drecera, tal com es veu a la següent imatge.



Il·lustració 8: Exemple de adició de una drecera en Contraction Hierarchies

Un cop fet el preprocés i construït el graf simplificat es realitza un dijkstra bidireccional del punt origen al punt destí.

3.3.2 Decisió sobre els algoritmes comentats

A continuació exposarem per que em escollit Dijkstra en comptes de les altres dos opcions:

	Dijkstra	**A***	**Contraction Hierarchies**
Complexitat	molt baixa	Baixa	alta
Cost exploració	Quadràtica (n^2)	lineal (n) millor cas quadràtic(n^2) en el pitjor	Quadràtic($(n-k)^2$) On k son nodes eliminats en el preprocés.
Cost de lectura de camí	lineal	lineal	lineal
Dependència heurística	no	si en l'exploració	si en preprocés

Taula 4: Comparativa d'algoritmes de camí més rapit

3.3.2.1 A* vs Dijkstra

A * és més ràpid que l'algorisme de Dijkstra,. Però la simplesa de Dijkstra i el fet que en el nostre cas(sense pesos negatius) té un cost quadràtic[10] el fan un algoritme suficientment eficient per el nostre problema.

[10] (27)Source: Dijkstra's Algorithm versus Uniform Cost Search or a Case Against Dijkstra's Algorithm, Ariel Felner

En conclusió, agafem Dijkstra en comptes de A* per que sent el primer suficientment potent no hem de dependre d'una heurística com en el cas de A*. Utilitzar una heuristica resultaria en perdre la certesa de que la solució que donem és la millor possible.

3.3.2.2 Contraction Hierarchies vs Dijkstra

Sobre Contraction Hierarchies contra Dijkstra ens trobem en un cas similar, en el cas de simplificació hem de recórrer a una heurística per eliminar nodes, el que pot donar resultats superiors a dijkstra. Però aquesta simplificació ens portaria una reducció en la precisió de la solució.

També és difícil justificar l'enorme cost de desenvolupar un Dijkstra (o+1)-direccional, on o correspon al nombre de punts d'origen possibles, si Dijkstra ja és capaç de fer l'exploració del graf en un temps acceptable.

4 Requisits

A partir de lo especificat fins aquest punt, podem extreure quines han de ser les característiques i funcions del sistema a desenvolupar per a poder complir amb els objectius que hem fixat. Aquests requisits suposaran el *backlog* d'aquest projecte, ja que fem servir la metodologia Scrum i diferenciarem entre requisits funcionals i no funcionals.

4.1 Requisits funcionals

Un requisit funcional defineix una funció del sistema de programari o els seus components. Una funció és descrita com un conjunt d'entrades, comportaments i sortides. Els requisits funcionals poden ser: càlculs, detalls tècnics, manipulació de dades i altres funcionalitats específiques que el sistema ha de complir.

En la taula següent podem veure els requisits funcionals relacionats amb l'objectiu de l'apartat 8.5 que ajuden a assolir.

Requisit	Objectius que assolim
El nou sistema ha de ser capaç de tenir les dades persistides en un format que permeti extreure una ruta i calcular el cost en temps d'accés, recursos i desnivell acumulat a peu.	4
El nou sistema ha de permetre a l'usuari demanar les possibles rutes per un rescat especificant un punt en concret mitjançant coordenades	1 i 4
El nou sistema ha de permetre els bombers decidir un punt de trobada	1
El nou sistema ha de permetre diferenciar les rutes i els seus temps d'accés	5 i 6
El nou sistema ha de permetre descartar rutes per tal de poder prendre una decisió	5 i 6
El sistema ha de ser capaç de generar una estructura que ens permeti explorar l'arbre de possibilitats de rutes minimitzant el cost en temps d'accés, recursos i desnivell acumulat a peu.	1 i 4
El sistema ha de ser capaç de permetre introduir diversos punts d'origen en diversos medis de transport	1 i 2
El sistema ha de ser capaç d'explorar l'arbre de possibilitats en el mínim cost en temps d'accés, recursos i desnivell acumulat a peu	1, 4

Taula 5: Requisits funcionals

4.2 Requisit no funcionals

Mentre que un requisit no funcional o atribut de qualitat és, en l'enginyeria de sistemes i l'enginyeria de programari, un requisit que especifica criteris que poden usar-se per jutjar l'operació d'un sistema en lloc dels seus comportaments específics.

En la taula següent podem veure els atributs de qualitat relacionats amb l'objectiu de l'apartat 8.5 que ajuden a assolir.

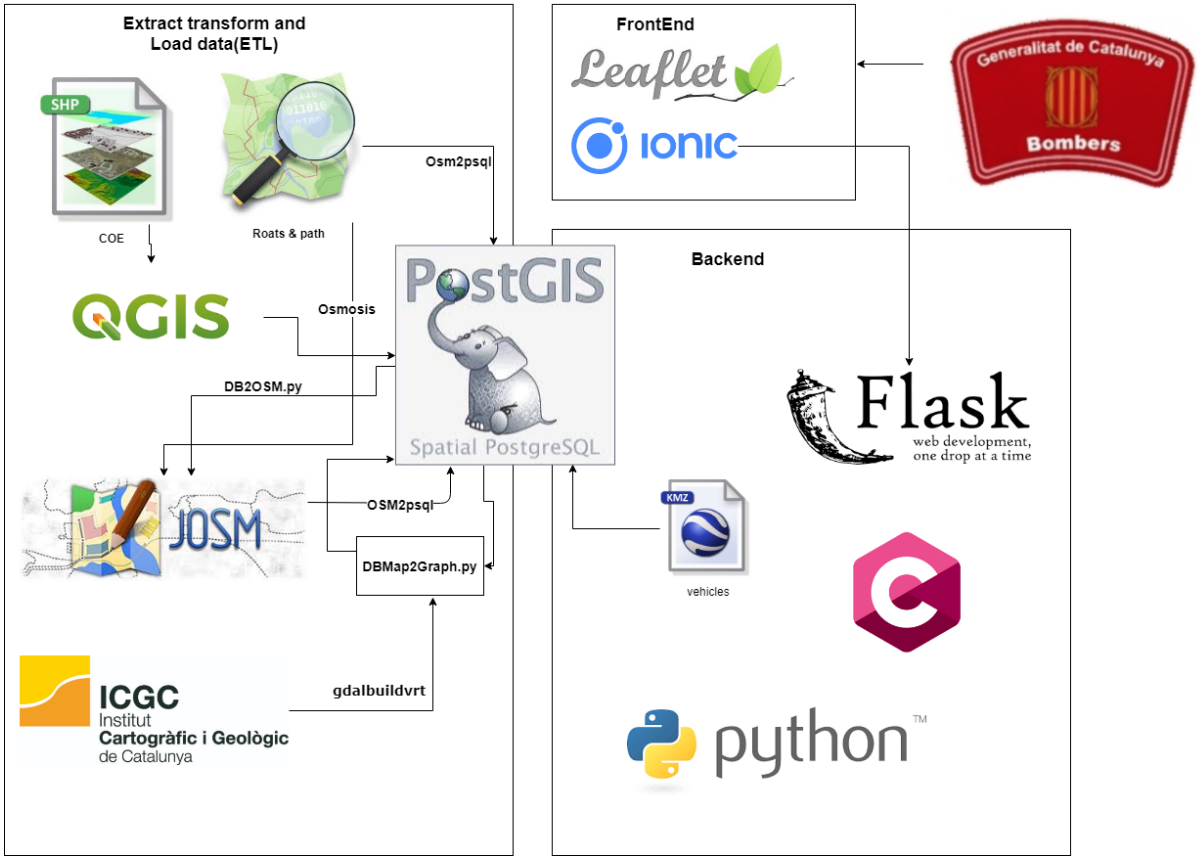
Requisit	Objectius que assolim
El nou sistema ha de comptar amb dades geogràfiques reals	1
El nou sistema ha d'integrar els camins COE	1 i 3
El nou sistema ha d'integrar les dades de vehicles dels bombers	2
El sistema ha de ser capaç de calcular distàncies, temps i diferència d'altures.	1
Els sistema ha de mostrar les rutes obtingudes per l'algoritme d'una forma clara i entenedora que li permet triar quina ruta escollir	4
El sistema ha de permetre introduir diversos punts d'origen i comparar les rutes obtingudes	1, 2 i 4
El sistema ha de permetre la configuració del posicionament dels recursos de bombers	1 i 2
El sistema ha de ha saber distingir entre els diferents tipus de via: carretera, pista, camí, sender, etc.	1 i 2
Les dades geogràfiques han de contenir les dades sobre l'altura	1
El sistema ha de fer distinció entre els diferents tipus de vehicle utilitzables: camió, tot terreny, i a peu.	2
El sistema ha de tenir un temps de resposta acceptable per un cas d'emergència 30s	1
S'ha de tenir en compte tots els punts possibles per a detectar la posició dels vehicles	1
El sistema ha de tenir en compte ser escalat a tot Catalunya	3
S'ha de tenir en compte tots els punts possibles per a detectar la posició dels vehicles	2
El sistema es desenvoluparà en Python	7
El disseny del sistema ha de tenir en compte un futur escalament del mateix	7

Taula 6: Atributs de qualitat

5 Tecnologies utilitzades

El següent diagrama mostra les diferents tecnologies implicades en el funcionament de l'eina de suport a la presa de decisions a l'hora de fer rescats de muntanya segons l'ús que se li dona.

Cadascuna d'elles es descriurà en les següents seccions i algunes d'elles es descriuen amb més detall en l'annex de Tecnologies d'aquesta memòria.



Il·lustració 9: Diferents tecnologies implicades en el funcionament del sistema

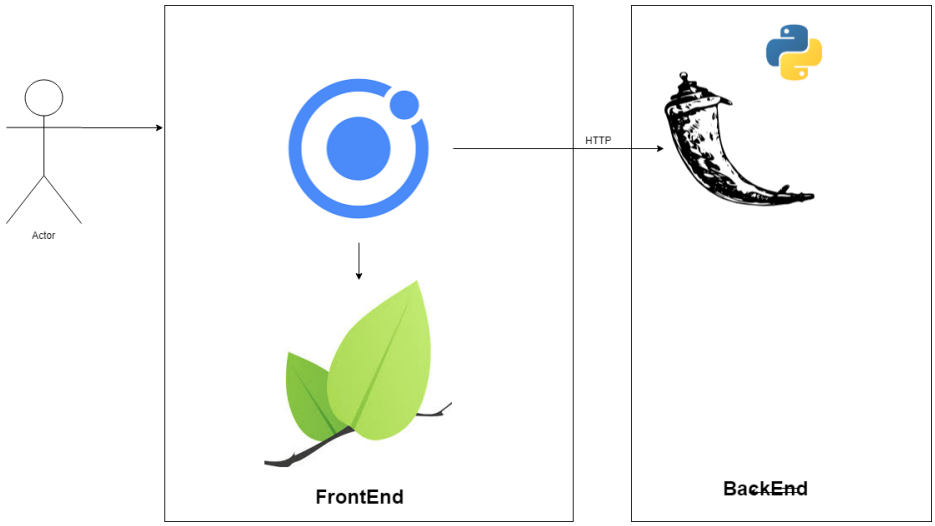
Les principals tecnologies implicades es poden agrupar en dues àrees IT:

- *Frameworks* i Llenguatges de desenvolupament
- Sistemes d'informació geogràfica

5.1 *Frameworks* i LLenguatges

A nivell de *Frameworks* s'ha utilitzat els següents frameworks:

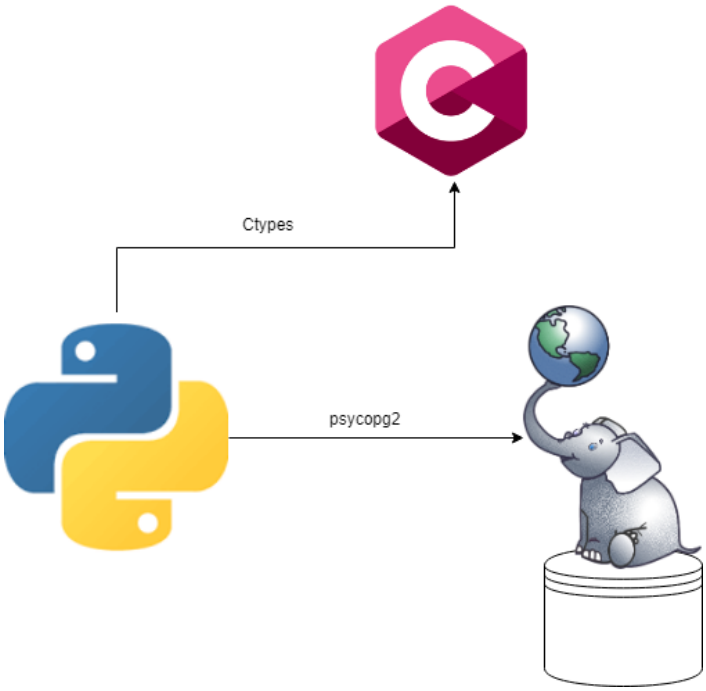
- **Ionic** (Cordova, Angular): Ionic es un framework construït sobre angular, ens permet escriure en el llenguatge Typescript (Javascript tipificat), codi que ens permetrà obtenir una aplicació mòbil per Android o iOS o també una web per navegador. S'ha decidit Ionic pel requisit de que la interfície ha d'estar implementada en un llenguatge que ens permeti un futur desplegament en mòbil. Juntament amb Ionic emprem la llibreria **Leaflet,** una llibreria *open-source* per fer mapes interactius *mobile-friendly*. S'ha decidit per la combinació de Ionic amb Leaflet per l'experiència previa
- **Flask**: És un microframework fet en Python, està classificat com a microframework ja que no requereix de cap altre llibreria o eina. La seva falta de dependències es el que ens va fer decidir per ell per resoldre les crides del *Frontend*



Il·lustració 10: Relació entre els frameworks

Dins del *Backend* s'han utilitzat els següents llenguatges de programació(també es recullen en l'annexa de tecnologies i llenguatges):

- **C**: Degut al requisit de que el sistema ha de ser capaç de calcular les rutes en un temps reduït, s'ha implementat una llibreria en c cridable des de python. Hem emprat c per la seva gran capacitat de càlcul.
- **Python**: L'ús d'aquest llenguatge ens permet integrar les diferents parts del codi, des de la gestió de la informació fins el càlcul de les rutes. Amb Ctypes em estat capaços de integrar-nos amb la llibreria de C, amb psycopg2 hem estat capaços de gestionar la connexió i consultes a la BD
- **pgsql:** L'ús d'aquest llenguatge ens permet fer enregistrar les dades en un format que ens permet fer cerques georeferenciades sobre una base de dades **postgresSQL.**



Il·lustració 11: Grafic de relacio entre llenguatges

Com es veu a l'anterior esquema per cridar des de python instruccions **pgsql** s'ha fet servir la llibreria **psycppg2** com la següent:

```
cursorEnconterPoints.execute("SELECT name, ST_AsKML(geom)"
  " FROM public.\"COE_montserrat_punts\""
  " where folderpath like '%Punt de trobada%' ;");
```

Il·lustració 12: Query des de python amb psycppg2

Com es veu a la il·lustració del gràfic de relació entre llenguatges s'ha fet servir **Ctypes** per emprar el codi en c. La llibreria **Ctypes** permet referenciar arxius de llenguatge C compilats com so(shared object library) per fer-los servir en python

Les il·lustracions 13, 14 i 15 mostren el procés per emprar ctypes.

```
extern "C" void inizializeGraph(int V);
extern "C" int findKey(long long nodeId);
extern "C" long long *findPredecesor(int key,int vehicle);
extern "C" double *findPredecesorValues(int key,int vehicle);
extern "C" void solve(long long start, int vehicle);
extern "C" void inicializeEdge(int nodeKey1,int nodeKey2, double walkingCost,double CarCost,
                    double  BRPCost,double AllTerrainCost,double rampPos, double rampNeg, double dist);
extern "C" int addNode(long long nodeKey);
```

Il·lustració 13: Capçaleres de les funcions en c

```
>gcc -c -fPIC dijkstraListCFile4.cc -o dijkstraListCFile3.o
>gcc -shared -W1,-soname,dijkstraListCFile3.so -o dijkstraListCFile3.so dijkstraListCFile3.o


>gcc -c -fPIC dijkstraMultimodalTotal.cc -o dijkstraMultimodalTotal.o
>gcc -shared -W1,-soname,dijkstraMultimodalTotal.so -o dijkstraMultimodalTotal.so dijkstraMultimodalTotal.o
```

Il·lustració 14: Crides de comanda per compila per Ctypes

```
myrouter = cdll.LoadLibrary('./RouteSelector/router/dijkstraListCFile3.so')
self.myrouter.inizializeGraph.restype = None
self.myrouter.inizializeGraph.argtypes = [c_int]
self.myrouter.inizializeGraph(numNodes)#execució
```

Il·lustració 15: Inicialització y execució de una llibreria emprant Ctypes

5.2 Sistemes d'informació geogràfica

Un sistema d'informació geogràfica (SIG o, en anglès, GIS) és un sistema informàtic, format per maquinari, programari, dades, usuaris i un marc organitzatiu, que permet enregistrar, emmagatzemar, gestionar, analitzar, consultar, visualitzar, presentar i difondre qualsevol tipus d'informació geoespacial.

La característica més rellevant i diferenciadora dels sistemes d'informació geogràfica és el fet d'emmagatzemar explícitament la posició geogràfica i la forma geomètrica de les entitats o fenòmens representats en el sistema d'informació.

Per tant un SIG ens dona la capacitat d'interrelacionar informacions d'entitats o fenòmens diferents per mitjà de la posició i la capacitat de realitzar operacions espacials amb les formes geomètriques de les entitats o fenòmens representats.

Les condicions indispensables per fer efectives aquestes capacitats úniques dels sistemes d'informació geogràfica són:

- La georeferenciació dels diferents conjunts de dades geoespacials, preferiblement segons un mateix sistema de referència espacial
- La representació espacial de la forma geomètrica de les entitats o fenòmens mitjançant models de dades espacials adequats com és ara el model de dades vectorial o el model de dades ràster.

Dins d'un SIG cal destacar els formats d'arxiu que es poden utilitzar i els productes específics que permeten la manipulació, consulta i operacions georeferenciades

5.2.1 Formats d'emmagatzematge d'informació geogràfica:

Els diferents productes per l'emmagatzematge de dades cartogràfiques que s'han emprat en el treball, són cinc, descrits amb més detall dins l'annex 'Tecnologies':

- **OSM**: Aquest format ens aporta una informació indexada que relaciona la connectivitat entre les dades geogràfiques, n'obtenim les dades de carreteres i camins externes al COE
- **KML**: Format en que els bombers guarden les últimes posicions dels vehicles
- **Shapefile**: Format en que els bombers tenen guardades les dades del COE
- **Raster**: Extret de les dades de l'Institut Cartogràfic i Geològic de Catalunya en format TXT ens aporta les altures que necessitem per complir el requisit de tenir les dades en 3 dimensions
- **VRT:** Format per la llibreria **GDAL** que permet fusionar i referencia dades de manera que GDAL les pugui emprar. El fem servir per referenciar les dades Raster
- **PostGis**: Aquest sistema ens resol el requisit de gestionar dades en 3 dimensions, el de tenir un format que ens permet gestionar la càrrega de dades del programa i ens

permet mantenir la informació indexada de manera que relaciona la connectivitat entre les dades geogràfiques

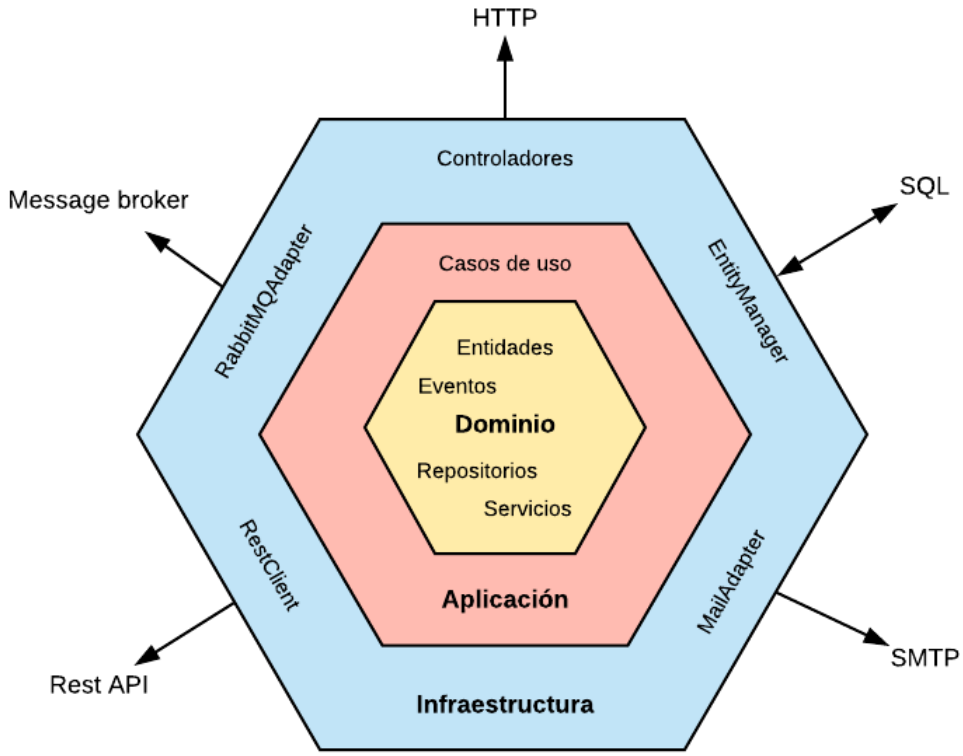
5.2.2 Productes cartogràfics

En el mercat podem trobar diferents productes comercials i de programari lliure que permeten transformar les dades en el format requerit en el treball. S'anomenen a continuació i són descrits amb més detall dins l'annex 'Tecnologies'

- **API Overpass:** Eina emprada per extreure dades del model del món de OSM
- **JOSM:** Eina que hem emprat per visualitzar les dades, comprovar que no hi hagin solapaments i arreglar solapaments e inconnexions validant així que les dades utilitzades siguin reals.
- **OSM2psql:** Eina que ens permet passar les dades de OSM a PostGis mantenint els avantatges de OSM
- **QGIS:** Eina que hem emprat per visualitzar les dades del COE i carregar-les a una base de dades PostGis
- **DB2OSM.py:** Script implementat en el marc d'aquest projecte. S'utilitza per extreure les dades del COE de PostGis en format OSM per poder ser validades i editades amb JOSM. S'explica en detall més endavant
- **web de ICGC:** S'utilitza per filtrar i descarregar dades raster en format text ASCII
- **gdalbuildvrt:** Eina per a construir els fitxers VRT a partir dels raster en **ASCII** això ens permet tenir un format que la llibreria de GDAL pot interpretar
- **Map2GraphDB.py:** Script implementat en el marc d'aquest projecte. S'utilitza per afegir les dades d'altures dels fitxers raster a la base de dades, i canviar el format de la base de dades per un que faciliti la generació i exploració del graf. S'explica endetall més endavant
- **OSMOSIS:** Eina emprada pel filtratge de dades OSM per a no sobrecarregar JOSM en la validació de dades ni en les primeres versions del sistema

5.3 Arquitectura hexagonal

La intenció de la arquitectura hexagonal no és cap altre que permetre que una aplicació sigui utilitzada de la mateixa forma per usuaris, programes, proves automatitzades o scripts, i sigui desenvolupada i provada de forma aïllada(*isolated*) dels eventuals dispositius i bases de dades.



Il·lustració 16: Esquema de un sistema hexagonal[11]

5.4 Servei web REST

Un servei web és defineix com una col·lecció de protocols i estàndards que serveix per intercanviar dades entre aplicacions. En el nostre cas es un servei REST, per tant fem servir el protocol HTTP per intercanviar la informació amb el *backend*.

Així doncs mentre el *frontend* s'executa a la maquina on estem treballant tenim un únic sistema que maneja tota la complexitat de càlcul. Aquesta estructura ens dona:

1. Escalabilitat. És més fàcil escalar el codi si es separa en Frontend i Backend. Hi ha diverses raons:.
    - Ja que el codi es divideix en dues parts, és pot optimitzar el codi més ràpidament.

[11] Source; revista digital Medium

- Es pot augmentar els recursos pel *frontend* i el *backend* a un ritme diferent, ja que el *backend* haurà de pujar a un ritme relativament més ràpid a mesura que es creix.
- Es pot afegir nous *frontends* que tractin les dades del Backend de maneres diferents, complint altres objectius

2. Optimització de recursos: Si és separa el codi entre Frontend i Backend els servidors només envien dades, en el nostre cas en format JSON, i es el navegador, que s'encarrega de gestionar les dades i treu aquesta lògica del servidor. Per tant els servidors han de treballar menys i l'experiència de l'usuari és millor.
3. Adaptabilitat al canvi. Mantenir el Backend i el Frontend per separat permet d'atacar els canvis de manera més simple, per parts.

6 ARM

ARM es un sistema pensat pel suport a la presa de decisions en el rescat de muntanya. Està compost per les següents tres parts que anirem explicant amb més detall en aquest apartat:

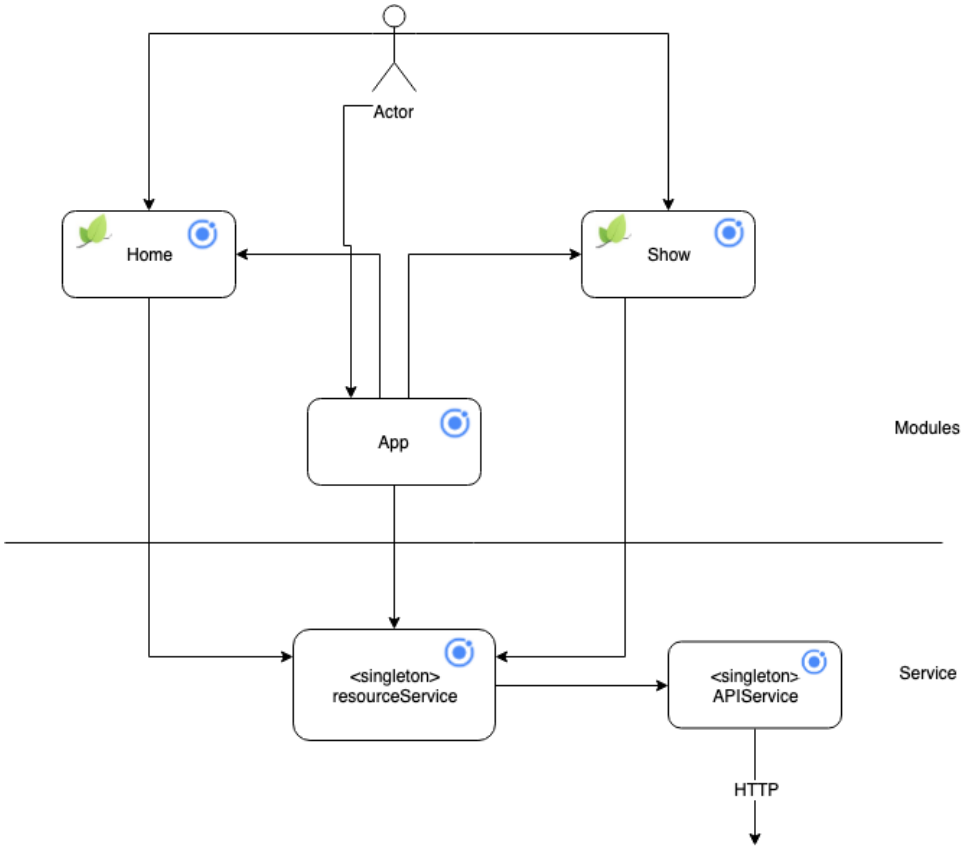
- **Frontend:** Interfície del sistema, capaç de visualitzar, ordenar i diferenciar possibles solucions per al per fer el rescat
- **Backend:** Gestiona el cas de trobar les millors rutes en el format d'aplicació web. Resolent les peticions que es fan des de Front End. Interpreta el fitxer de ultimes posicions, explora la cartografia retornant les rutes.
- **ELVT:** Procés d'extracció, càrrega validació i transformació que s'empra per posar a punt la base dades,.

6.1 Frontend

La funcionalitat principal del Frontend és la interacció amb l'Usuari final i per tant gestiona tots els casos d'us que estan relacionats amb la visualització de dades,

Per entendre com és la estructura del Frontend, el dividim en dos:

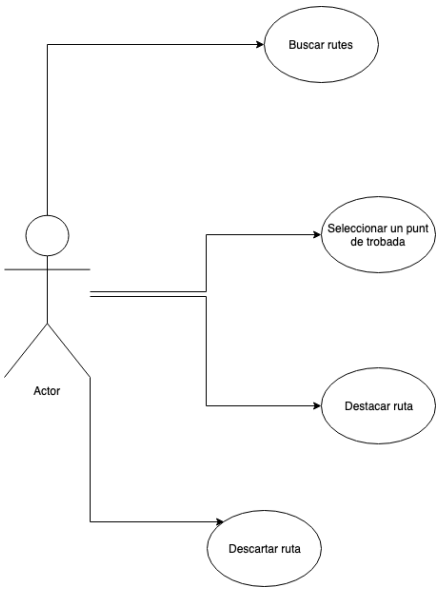
- **Mòduls:** Gestionen la interfície d'usuari, en tenim tres:
    - **App:** Component principal d'Angular que conte els altres dos mòduls. Controla el menú que ens permet seleccionar si volem veure les rutes de un punt de trobada en concret o fer una nova petició
    - **Home:** Gestiona la interacció de l'usuari per fer la petició per obtenir les rutes a partir de les coordenades de rescat
    - **Show:** Mostra a l'usuari les rutes possibles fins al punt de rescat per un punt de trobada.
- **Serveis:** Gestionen l'estructura de dades fent els càlculs i crides pertinents, en diferenciem dos:
    - **resourceService:** Gestiona el recurs, cridant a la API Service per actualitzar la informació sobre les rutes tant bon punt es resol la *promise* que ens retornarà la **APIService**
    - **APIService:** Demana el resultat a la Api via HTTP, i retorna una *promise.* que en resoldre's tindrà el resultat de la crida.



Il·lustració 17: Esquema de la estructura del Frontend

6.1.1 Casos d'ús

Seguidament explicarem els diferents casos d'ús que resol el *frontend*, aquests tenen com a únic actor l'usuari final, els membres del Cos de Bombers de la Generalitat de Catalunya:

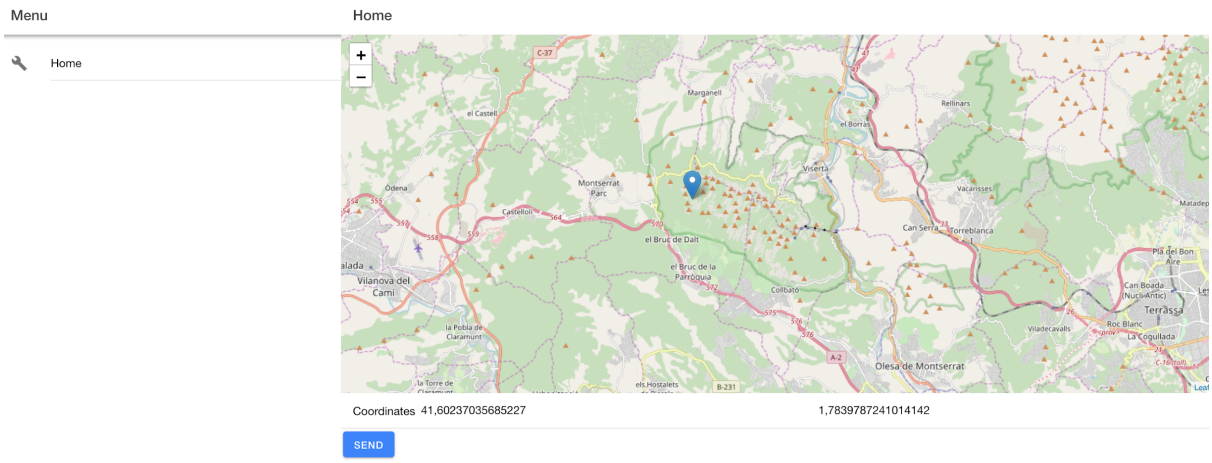


Il·lustració 18: Diagrama de casos d'us del Frontend

6.1.1.1 Buscar rutes

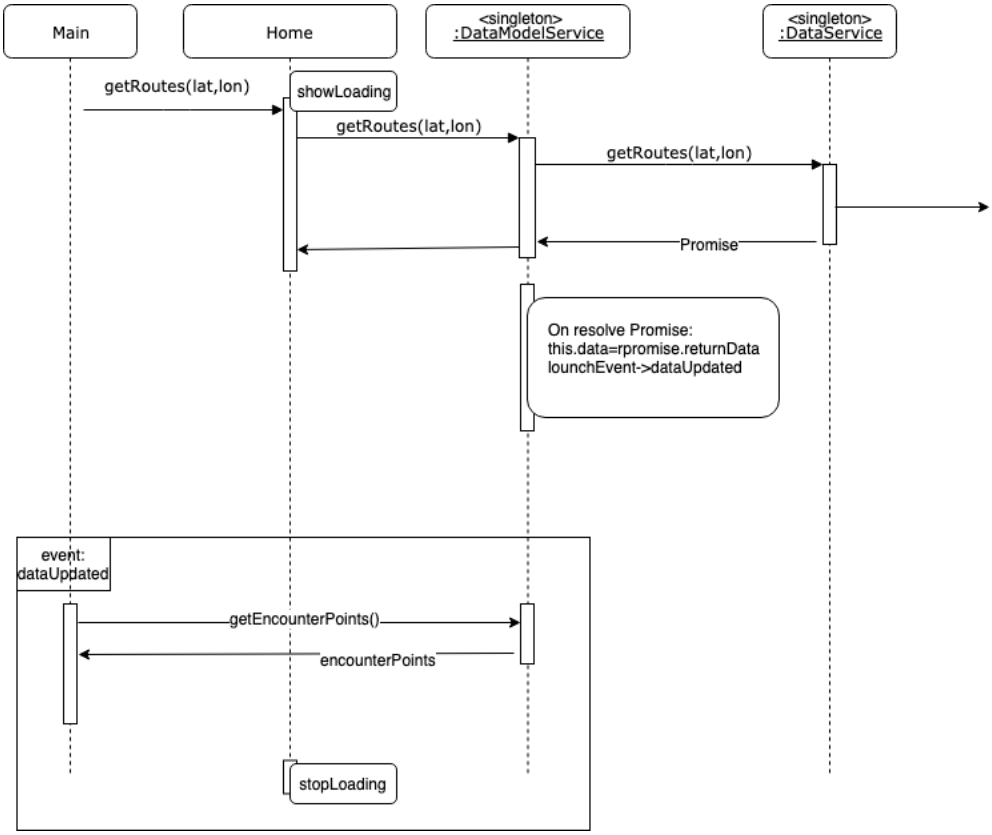
Aquest cas d'us permet a l'usuari demanar les possibles rutes per un rescat especificant un punt concret mitjançant coordenades.

Això és pot fer en la primera finestra que es mostra al *frontend* corresponent la següent il·lustració on es permet detallar el lloc de l'accident i demanar les rutes fins allà. El punt es pot incloure tant introduint les coordenades a mà com clicant en el punt sobre el mapa.



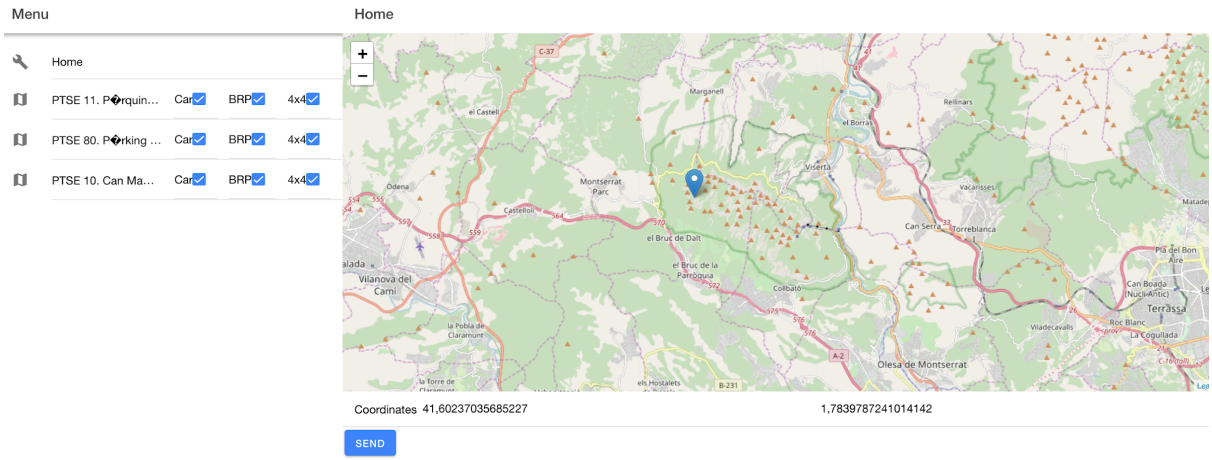
Il·lustració 19: Menu de selecció del punt de rescat

Un cop es pulsa el botó d'enviar, el controlador de la vista Home, que controla el formulari li demana al DataModelService que s'actualitzi, fent la peticio al *backend* a partir del DataService tal i com es mostra en el diagrama de seqüencia següent



Il·lustració 20: Diagrama de seqüencia del cas d'us demanar rutes del Frontend

Un cop el *Main* té els punts de trobada, aquests es mostren en el menú lateral, obtenint un resultat tal i com es veu a la següent imatge.

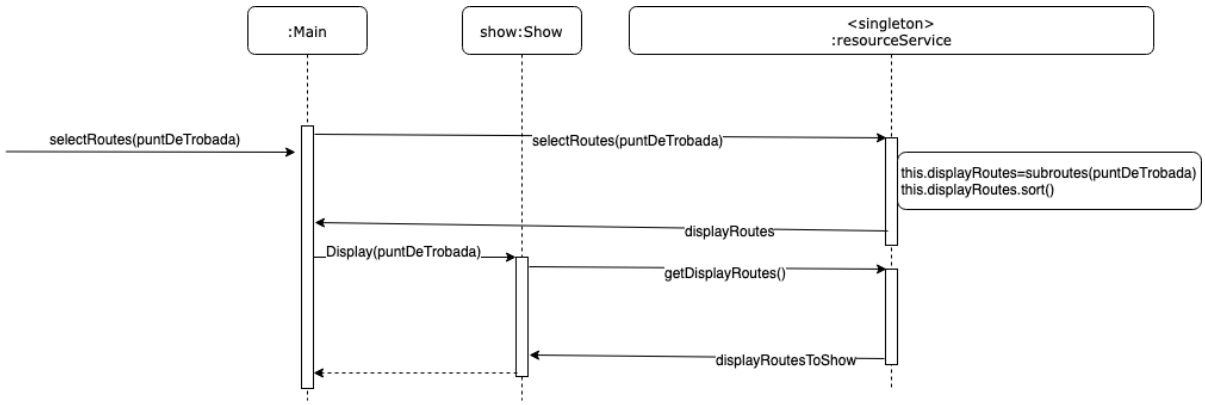


Il·lustració 21: Captura de pantalla un cop els camins han estat calculats pel Backend

6.1.1.2 Seleccionar un punt de trobada

Com s'ha mencionat anteriorment el nou sistema ha de permetre els bombers decidir un punt de trobada. Això és pot fer des de el menú, un cop resolta la petició pel punt de rescat.

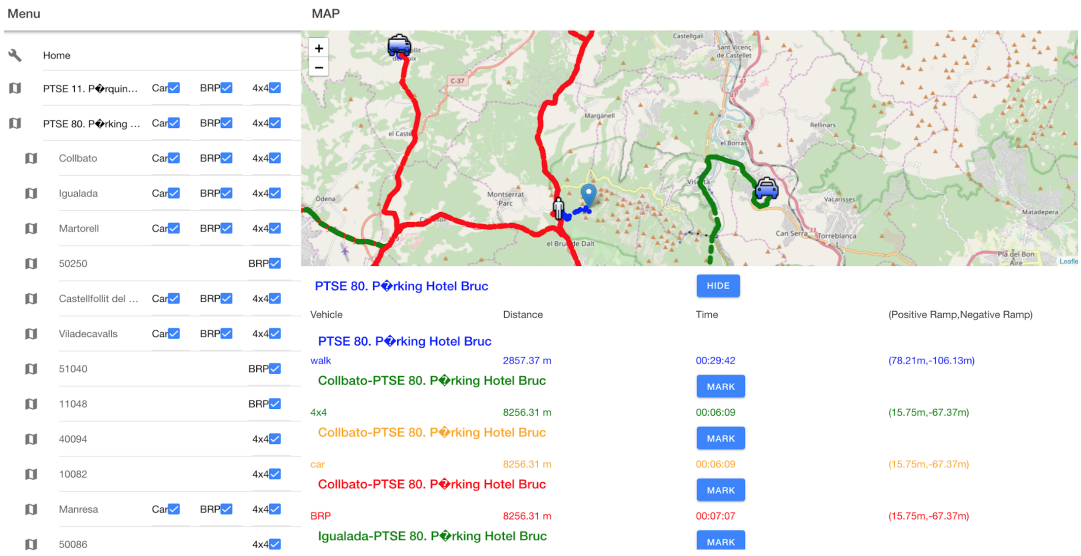
Des del menú podem seleccionar qualsevol de les opcions, que és corresponen al millors punts de trobada més propers trobats, per poder visualitzar les rutes fent servir el punt de trobada seleccionat, acte seguit el controlador de *Main* que gestiona el menú i del que hereten les demes vistes, demanarà les subrutes per mostrar-les i un cop les té redireccionarà cap a mostrar el mapa, amb les rutes decidides, tal i com es mostra en el diagrama de seqüencia següent.



Il·lustració 22: Diagrama de sequencia del cas d'us del Frontend seleccionar un punt de trobada

Un cop el controlador de *Show* ha completat la seva tasca, la vista que es veu és correspont amb la imatge següent. On es mostren les rutes, amb el orígens indicats tant al menú com a la secció d'informació de la pàgina. En amdos llocs apareixen en el mateix ordre, de més ràpid a més lent.

En la finestra de informació de les rutes des del punt de Trobada, primer surt la infromació de la ruta del punt de trobada al punt de rescat, i seguidament totes les rutes dels vehicles fins els punts de rescat. Es pot amagar i mostrar la finestra d'informació de la ruta amb el botó *Hide/Show*.

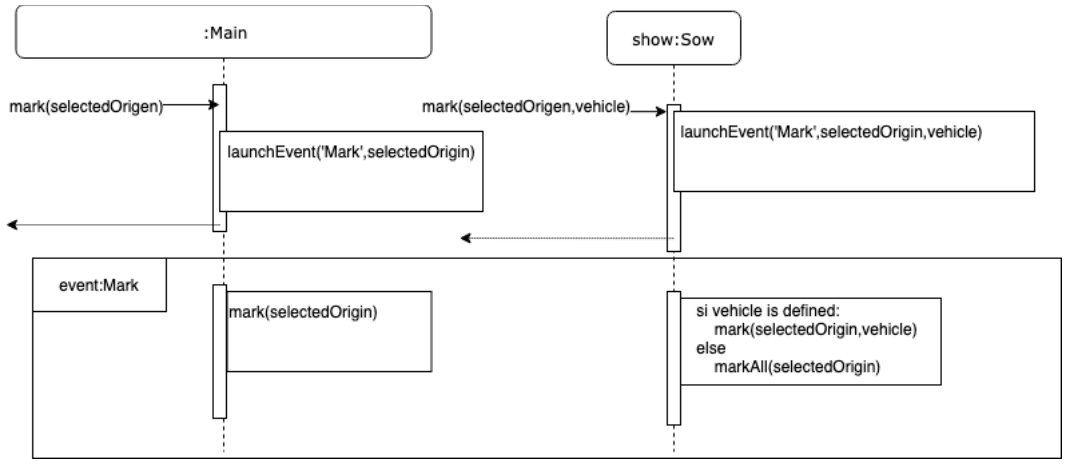


Il·lustració 23: captura de la pantalla de mostrar les rutes per un punt de trobada

6.1.1.3 Destacar ruta

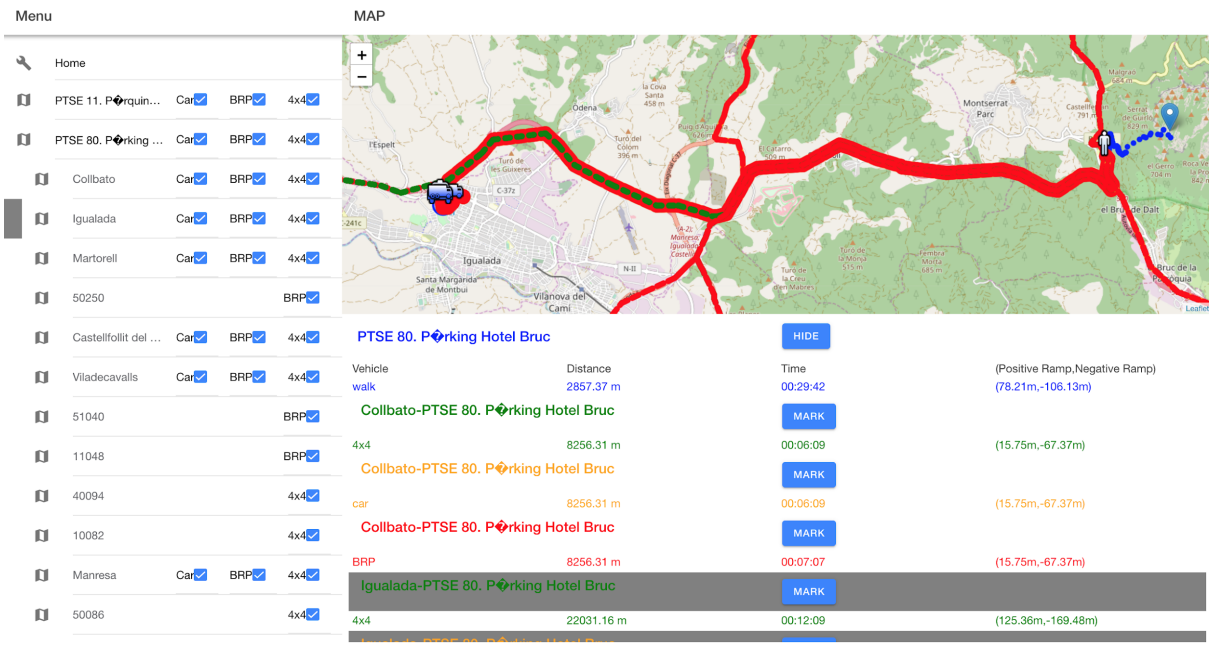
Com el nou sistema ha de permetre diferenciar les rutes i els seus temps d'accés, s'ha implementat aquesta funcionalitat que permet destacar una ruta en concret

Això és fa des de la vista del mapa *Show*. Tant des de la finestra on es mostren les dades de les rutes i des del menú es poden destacar rutes, clicant en el menú al nom del punt de origen o al botó *MARK* a la finestra d'informació. La lògica del procés de marcar ve indicada en el diagrama de seqüencia següent:



Il·lustració 24: Diagrama de seqüencia del cas d'us de frontend marcar ruta

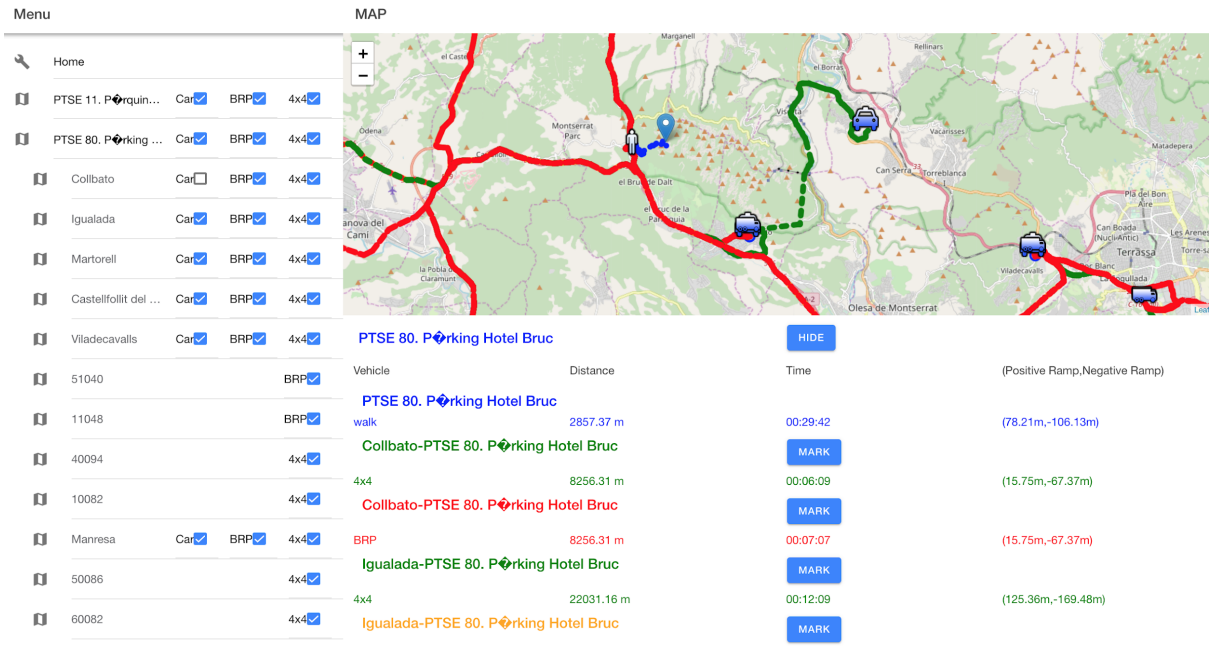
Si per exemple destaquem les rutes de totes les opcions dels vehicles del parc de bombers de Collbató el resultat serà com el que es veu a la següent imatge



Il·lustració 25: captura de pantalla de una ruta selecionada

6.1.1.4 Descartar ruta

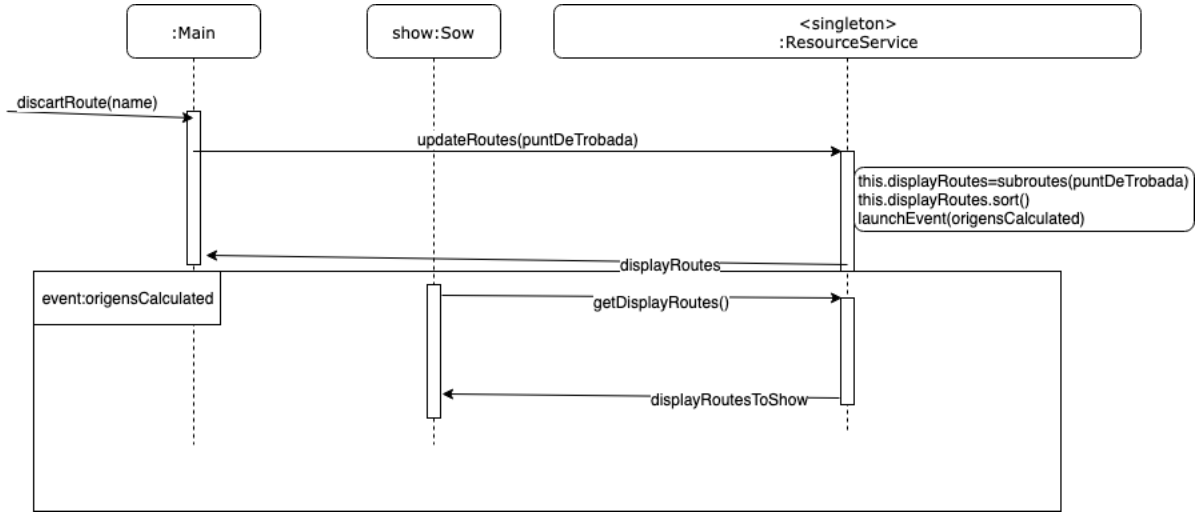
El nou sistema permet descartar rutes per tal de poder prendre una decisió, en fer-ho la ruta ja no es mostrarà a la finestra de mostrar rutes i baixarà a sota de tot en el menú, tal i com es veu en la imatge següent havent descartat la ruta en cotxe des de Collbató i el BRP 50250:



Il·lustració 26: Catura de pantalla amb rutes amagades

En aquest cas el controlador del menú envía la llista de selectedRoutes al DataModelService, extraient els orígens que no s'hauria de mostrar, un cop calculat el DataModelService llença

l'event origensCalculated i la finestra de mostrar rutes respon al event actualitzant les rutes a mostrar tal i com es mostra al següent diagrama de seqüencia.

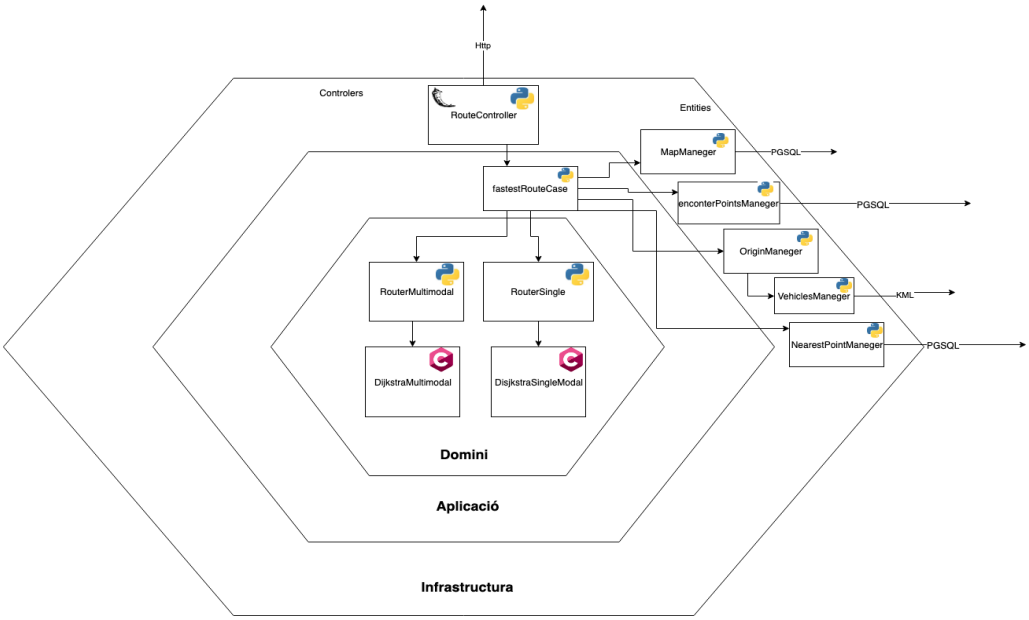


Il·lustració 27: Diagrama de seqüencia del cas d'us amagar ruta

6.2 Backend

La funcionalitat principal del *backend* és la de calcular les rutes, resolent les crides que es fan des de el *frondend*.

S'ha intentat mantenir la lògica del cas d'ús segregat de la capa de domini, i a la seva banda segregat de la capa de infraestructura. Intentant mantenir la idea d'un sistema hexagonal. Com es veu a la imatge següent:



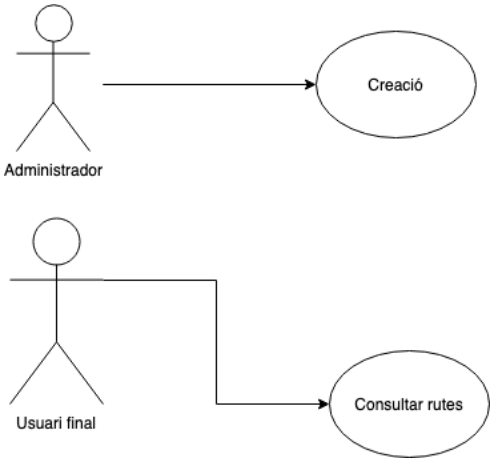
Il·lustració 28: Esquema de l'arquitectura del Backend

Un dels avantatges d'aquesta arquitectura és la seva escalabilitat, fruit de la gran independència entre les diferents parts de manera que hem pogut canviar i afegir les fonts de dades contínuament durant tot el projecte. Permetent-nos anar d'un sistema molt simple al sistema actual.

6.2.1 Casos d'ús

Els casos d'ús pel *backend* tracten el càlcul de les rutes i la gestió de les seva informació. Els que portaran a terme aquestes tasques son dos:

- **Administrador:** Muntarà el sistema com ho especifica el document README
- **Usuari final:** Els membres del cos de Bombers que interaccionaran amb el *backend* a traves del *frontend*

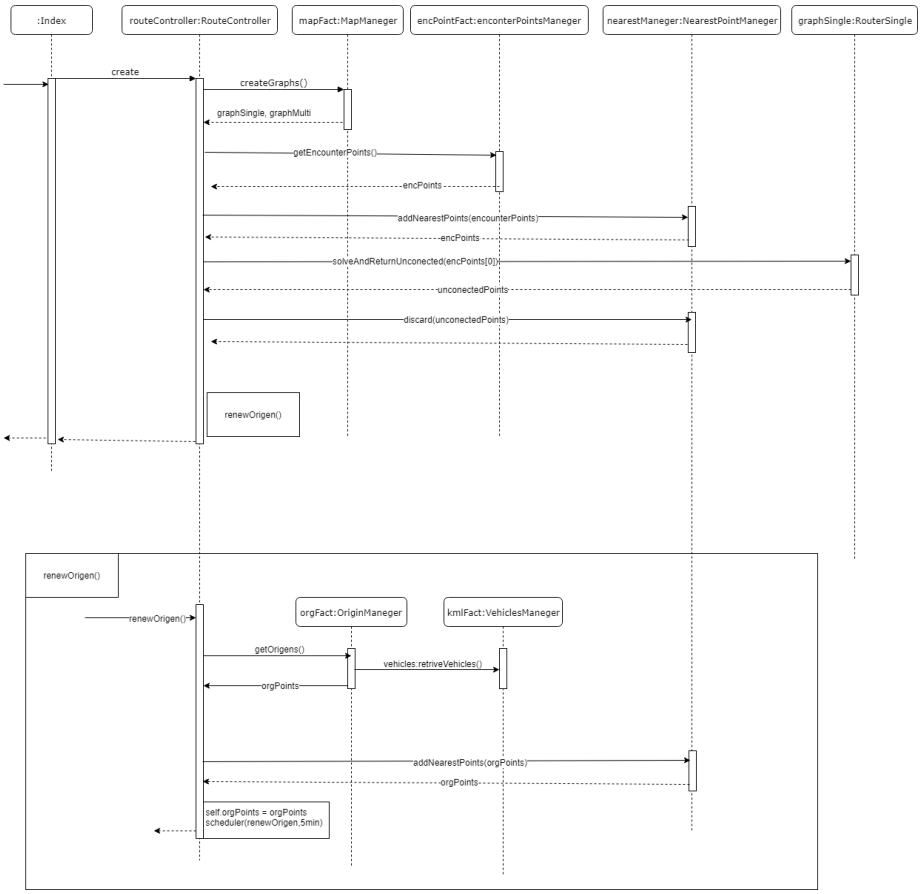


Il·lustració 29: Diagrama de casos d'ús del backend

6.2.1.1 Creació

El cas de creació és el cas on el servidor s'ha de llençar i a partir de les dades obtingudes de base de dades i un preprocessament addicional, s'obtindran dos grafs a memòria que permetran accelerar les consultes necessàries per a obtenir les diferents rutes que donarà el nostre DSS.

La seqüencia del cas d'ús en concret ve definida en el diagrama de seqüencia següent:



Il·lustració 30: Diagrama de seqüència del cas d'ús de creació del backend

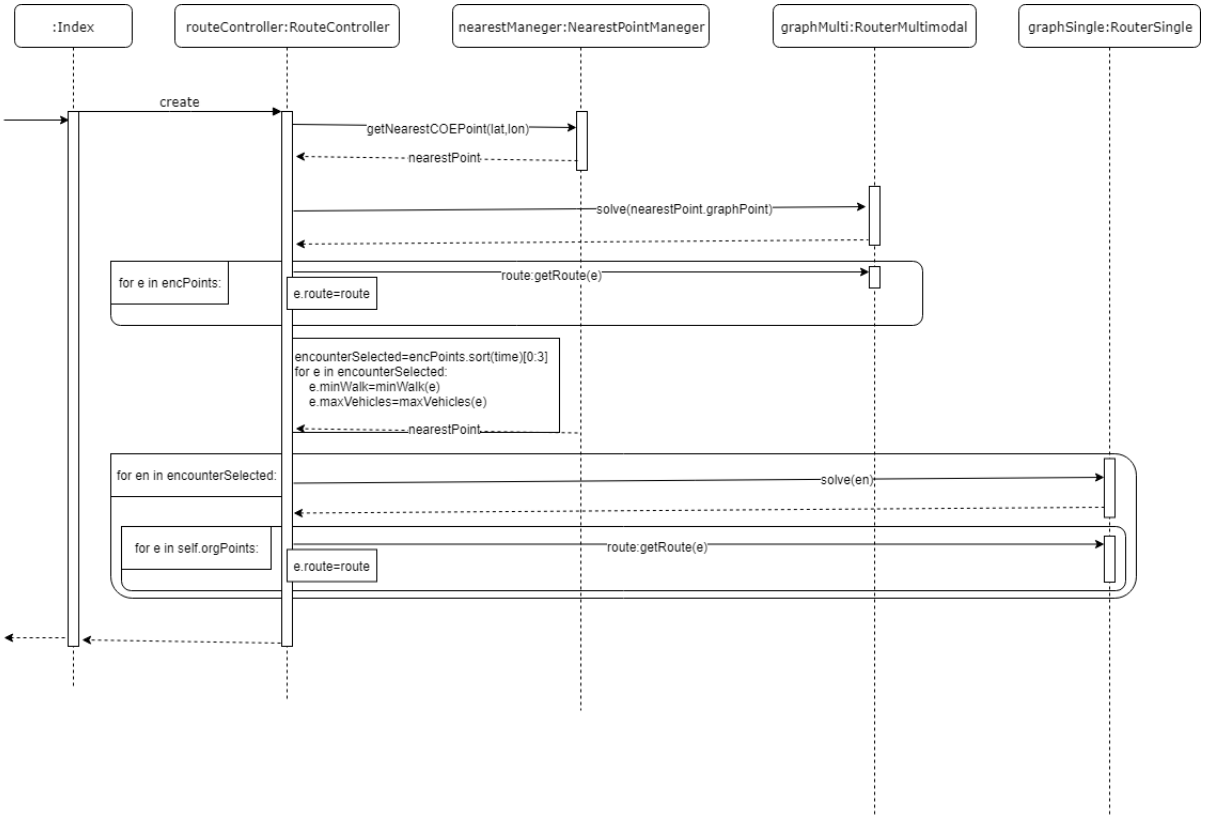
Del diagrama de seqüència anterior, cada classe té una responsabilitat:

- **MapManager:** Demana les dades a base de dades i genera el graf tornant dos instancies una de graf multimodal i una de singleModal que ens permetran explorar les rutes
- **EncpounterPointManager:** Retorna els punts de trobada
- **RouterSingle:** Permet explorar el graf i detectar quins nodes no són connexes des dels punts de trobada
- **NearestPointManager:** Permet descartar els punts no connexes i identificar un seguit de punts de la realitat, amb els punts del graf. Empra dos taules, ambdues taules estan ordenades per un índex **GIS** que ens permet fer més eficient una unió per ***kNN*** *(k-Nearest Neighbor).* Aquestes dos eines ens les dona ***Postgis.*** Les des taules són:
    - **all_nodes_geometry** que conté tots els punts dels camins navegables pel graf i quin són els punts del graf més propers
    - **temp_positions_geometry** que conté les posicions de tots els punts que és volen trobar.

6.2.1.2 Consultar rutes

Aquest cas comença quan el *frontend* demana unes coordenades, per això el *backend* realitza les següents accions:

1. Resoldre les peticions que es fan des de *frontend*.
2. Explorar la cartografia COE amb les rutes fins els punts de trobada.
3. Calcula els 3 millors punts de trobada possibles per realitzar el rescat
4. Calcula els sub-punts de trobada (fins on pot arribar cada vehicle en qüestió)
5. Calcular els accessos als punt de rescat per tots els vehicles fins els punt de trobada en qüestió mitjançant la cartografia OSM.



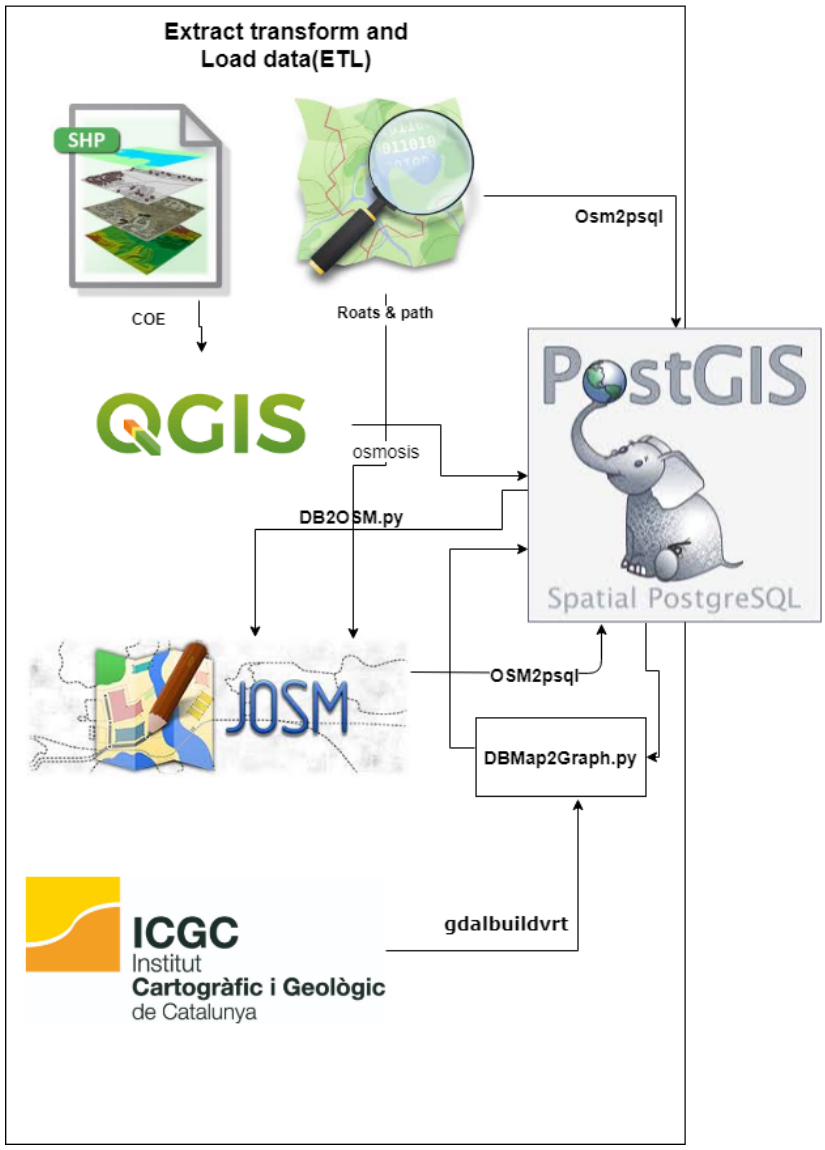
Il·lustració 31: Diagrama de seqüencia del cas d'us consultar rutes del Backend

6.3 Extract Load Validate Transform (ELVT)

El procés Extract Load Validate Transform és una variant de Extract Transform and Load (ETL). Aquest procés és necessari ja que com s'ha dit en l'apart anterior, el *backend* ha de ser capaç de decidir rutes extretes de la cartografia real. Les dades del COE es troben en un format que no permet la seva navegació sense un tractament, a més les dades de bombers només comprenen el Parc Natural. Necessitem més dades que les del COE per que els vehicles que es troben fora del parc puguin accedir al punt de rescat.

Per fer això complint el requisit abans mencionat, es fan servir dades del model OSM que tot i que compleixen el requisit de que relacionen l'accessibilitat entre les dades geogràfiques, no compleixen el de tenir informació sobre l'altura.

Per resoldre tot això, es fa un procés que extreu les dades de la cartografia de bombers i OSM i el posa en un format que els generadors d'entitat del *backend* saben interpretar.
Tal i com es mostra en el següent graf.



Il·lustració 32: Esquema de tecnologies implicades el el proces de ELVT

Aquest procés es compon de tres passos:

1. Extracció de la informació i persistència
2. Validació
3. Transformació del subconjunt de dades que cal adaptar al sistema de representació

6.3.1 Extracció i persistència

Com s'ha vist en l'apartat anterior tenim diverses fonts de dades, les que ens interessa per aquest procés són la **cartografia operativa de Bombers** i les dades extretes de **OSM** emprant la **API Overpass.**

6.3.1.1 Cartografia Operativa d'Emergències(COE)

En format *shapeFile*, en podem distingir tres fitxers principals la resta són metadades que complementen la informació d'aquests:

- **COE_montserrat_linees:** aquest fitxer conté els camins i pistes que podem fer servir dins el parc de Montserrat. Podem observar que amb la etiqueta folderpath podem veure si el camí és de la categoria **Camins i Pistes** o **Senders.** També podem veure que la etiqueta *name* fa referència al tipus de camí o pista. Aquests tipus de camins i pistes i les seves implicacions les trobem al manual d'inventari camins Bombers.
- **COE_Montserrat_punts**: podem trobar, fonts, camins tancats al públic, punts de trobada, etc. En concret ens interesen els **Punts de trobada,** aquests estan definits com el punts màxims on et pots endinsar dins al parc de Montserrat en qualsevol vehicle.
- **COE_Montserrat_poligons:** Conte les delimitacions de les diferents àrees de Catalunya pel cos de bombers. Per aquest treball no el farem servir.

Aquestes dades les obrirem amb QGIS on podem validar que el format es l'adequat i les abocarem dins la base de dades mitjançant la mateixa eina.

6.3.1.2 Zona de Catalunya del model del món OSM

Dades extretes de OSM emprant la API Overpass, en format OSM, conté tant la xarxa de camins i carreteres, com tot de punts d'interès. Tota aquesta informació la abocarem a la base de dades emprant l'eina **OSM2psql** amb la opció *slim* que ens ajuda a mantenir l'estructura de OSM. Aquestes dades també seran filtrades per nomes mantenir les dades de pistes i carreteres amb l'eina **OSMOSIS,** aquest pas el portarem a terme per poder validar la unió de les dades amb l'eina **JOSM** en el següent pas.

6.3.2 Validació

Per poder fer la primera validació de les dades hem de transformar les dades del **COE** al format **OSM,** que té un format navegable. Això ho farem amb l'script **DB2OSM** que hem desenvolupat amb aquest propòsit.

DB2OSM agafa els punts de trobada els aboca en el fitxer OSM, agafa els camins, i els camps nom i folderpath de cada camí, que es necessitaran a l'hora de decidir de quin tipus és el camí pel graf.

Les dades que es validen son les del **COE**, es **comprova** que els camins de DB2OSM no estiguin indegudament no connectats i s'ajunten en cas de que sigui així fent servir les eines que ens dona **JOSM**. Aquesta validació és complementa amb l'script desenvolupat **nodesDesconectats.** Aquest script explora el graf i dona els diferents conjunts connexos.

També és important comprovar que la posició dels punts de trobada del **COE** en el model OSM, per la definició de punt de trobada, es troba o bé a la punta (inici o final) de un *way* de OSM per on pot transitar tots els vehicles, o en una cruïlla de un Way per on circulen tots els vehicles i un *way* que no. De no ser així es partirà el *way* per on circulen tots els vehicles i ens permet arribar fins al punt de manera que compleixi aquesta condició.

Com en el procés anterior em filtrat les dades de carreteres i pistes de **OSM,** podem afegir aquestes dades al **JOSM** per comprovar i seguidament forçar aquesta propietat directament a la base de dades.

6.3.3 Transformació

Un cop arribat a aquest punt podem comprovar que una gran majoria dels punts només fan de nexe entre dos nodes. De manera que si fem servir aquest punt a la ruta sabem que és per anar al node següent.

A partir d'aquesta premissa podem simplificar el nostre model fent un preprocés en les dades de **OSM.** La idea seria aconseguir simplificar el graf amb un preprocés tal i com fan els sistemes basats en **Jerarquies de contracció**.

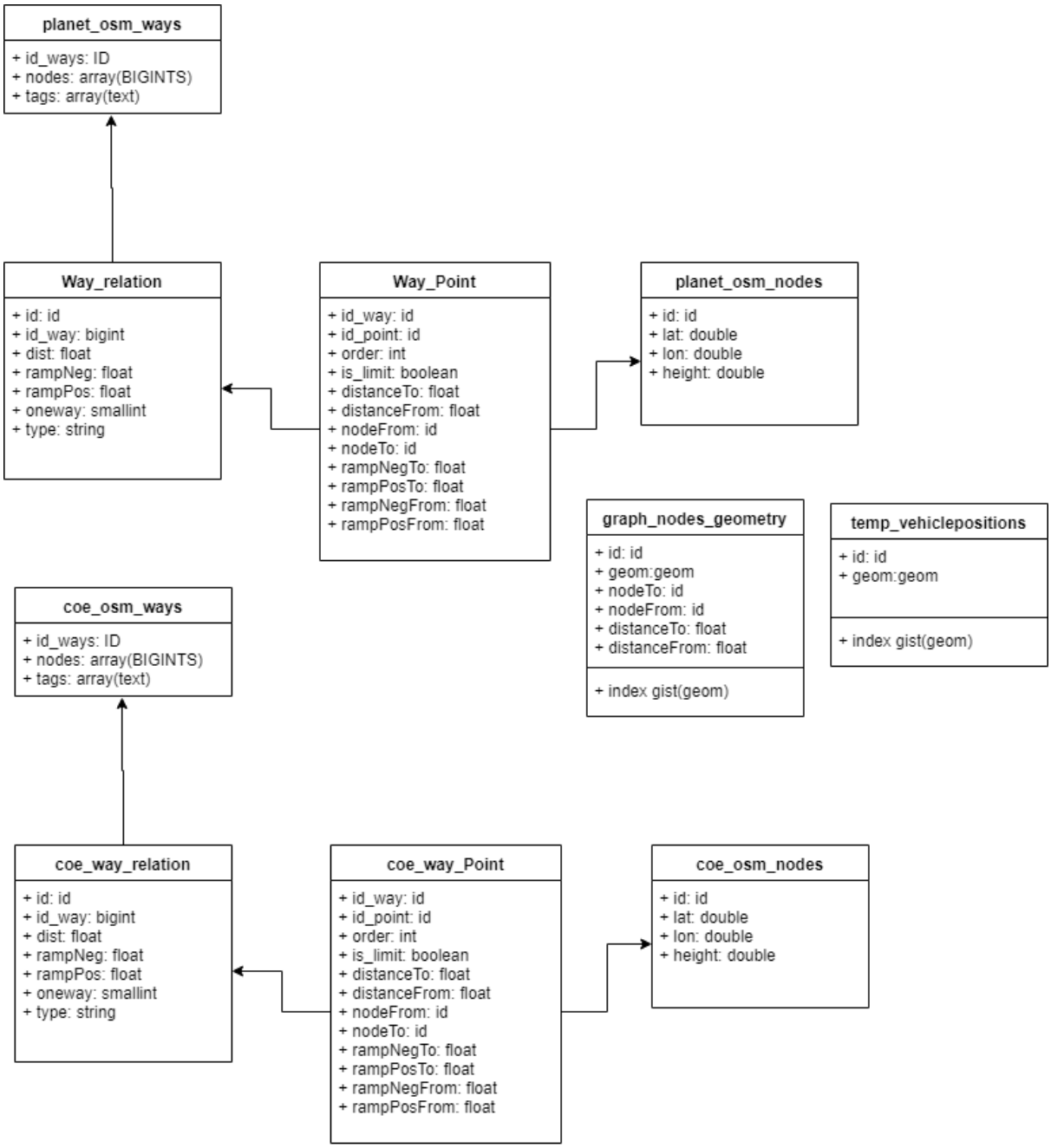
L'objectiu d'aquest preprocés serà eliminar del graf tots aquests punts que no aporten informació i conseqüentment són una simplificació molt important pels graf resultants del model **OSM**. El més important: l'eliminació de nodes no afecta l'accessibilitat dels punts de trobada per la particularitat d'aquests explicada en el l'apartat anterior.

Una característica addicional al procés de transformació és que és immutable a l'estat de la base de dades, per tant tot i que aconseguim eliminar els nodes del graf no perdem la resta de punts, ja que tot i que no formen part del graf segueixen estant en persistits i relacionats amb els punts que són del graf.

Aquesta tasca la realitza el script **storeGraphToDB**, primerament afegeix les altures a la base de dades, ja que per calcular les distàncies les hem de tenir en compte, seguidament calcula

tots els punts de creuament entre camins (qualsevol punt que apareix en dos camins), i talla tots els camins perquè com a màxim en tinguin dos. Un cop fet això es calculen les distàncies de tots els *ways*, guardant en els nodes intermitjos les dos distàncies fins al límit.

El diagrama UML següent mostra l'estructura de la base de dades que omple aquest script, i podem veure tant l'estructura per les dades del COE com les del model OSM.



Il·lustració 33: Diagrama estructural de la base de dades

També hi ha les taules ***graph_nodes_geometry*** què hi mantenim ordenades amb un índex de PostGIS els punts del mapa connexos en el graf, que si que s'omple amb aquest script, però el backend en treure els punts inconnexos i temp_positions_geometry on hi mantenim els punts dels vehicles també de forma ordenada.

Aquestes dos taules ens permeten reduir el temps de càlcul de les posicions dels vehicles dins el graf. Passem de fer 220 crides per cercar per cada vehicle quin és el punt més proper dels 3984393 que tardaven de l'ordre de 2 minuts a fer una Join kNN entre les dos taules, que juntament amb la lectura del fitxer triga **0.19s**

Amb aquest procés de transformació es passa de 4101219 arestes a 472781 és a dir només ens quedem un **11.5%** de les **arestes** originals i de 3984393 nodes a 373169 representant al graf només un **9%** dels **nodes** originals en la cartografia extreta de OSM.

Per altra banda a la cartografia COE es passa de 33757 a 1103 arestes és a dir, es simplifica el nombre d'arestes a un **3%** del original i un 32707 a 923 es redueix a un **2.8%** del nombre de nodes original.

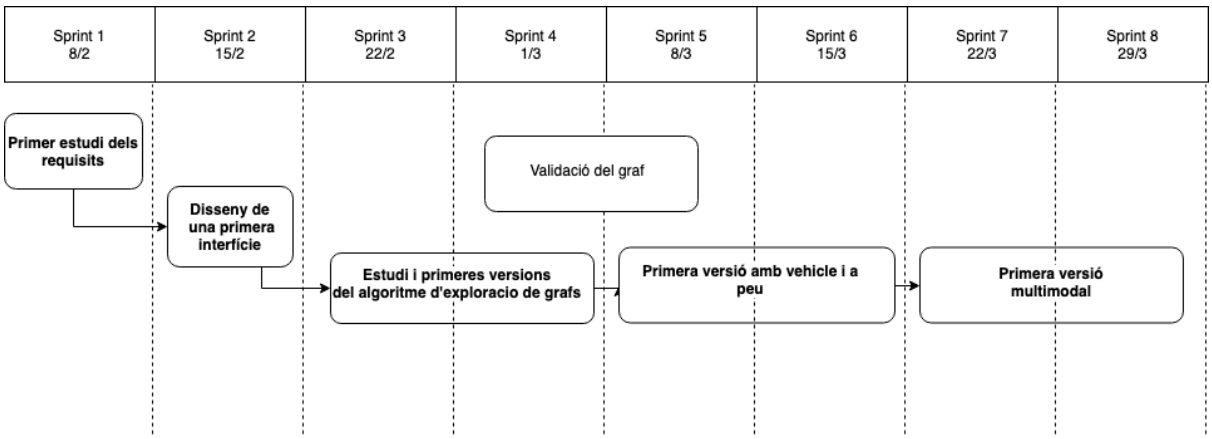
7 Evolució del Projecte

En les figura següents es mostren l'evolució del projecte. Entenem que el projecte es desenvolupa en tres fases:

- **Primera versió:** En la primera fase s'extreuen els requisits i es desenvolupar una primera versió que combini els diferents vehicles
- **Optimització de la cerca:** En aquesta segona fase s'amplia el mapa i s'optimitza l'algoritme en temps a l'hora de resoldre les possibles rutes
- **Optimització de precarga:** Es treballa en cartografia real, augmentant la mida del graf i del Mapa, i es millora el sistema de preprocés i carga

7.1 Primera versió

En la primera fase s'extreuen els requisits a partir de la primera reunió amb els bombers i es va desenvolupar una primera versió que combines els diferents vehicles i ens permetés la seva representació, treballant en una cartografia reduïda.



Il·lustració 34: Diagrama de Gant de la fase primera versió

7.1.1 Primera reunió amb els bombers

Es va fer una primera reunió amb tot l'equip i d'aquesta se'n va extreure que, els bombers, tenien la necessitat d'una eina de suport per trobar la ruta òptima considerant factors com el tipus de via, desnivell del terreny o els vehicles utilitzats i els camins de retorn en el parc de Montserrat ja que, actualment, la decisió de quina ruta es segueix es pren, únicament, mitjançant l'experiència pràctica dels bombers. De manera que no hi ha cap sistema que validi

que el camí decidit sigui l'idoni i provoca que hi hagi una excessiva dependència dels bombers més experimentats.

En aquesta primera reunió els bombers van explicar que una de les eines que feien servir molt era el mòbil ja que visualitzen la seva cartografia amb la plataforma OruxMaps i, alhora, empraven Google Maps per la seva capacitat de decidir rutes -tot i que aquesta aplicació no contempla tots el possibles camins-

També es va parlar de com estaven estructurats els camins de la cartografia dels bombers, i de que disposaven d'un fitxer -que s'actualitza cada pocs minuts- amb les últimes posicions dels seus vehicles. És a dir, indica on estan els vehicles en el moment de la darrera actualització.

7.1.2 Sprint 1: Primer estudi dels requisits

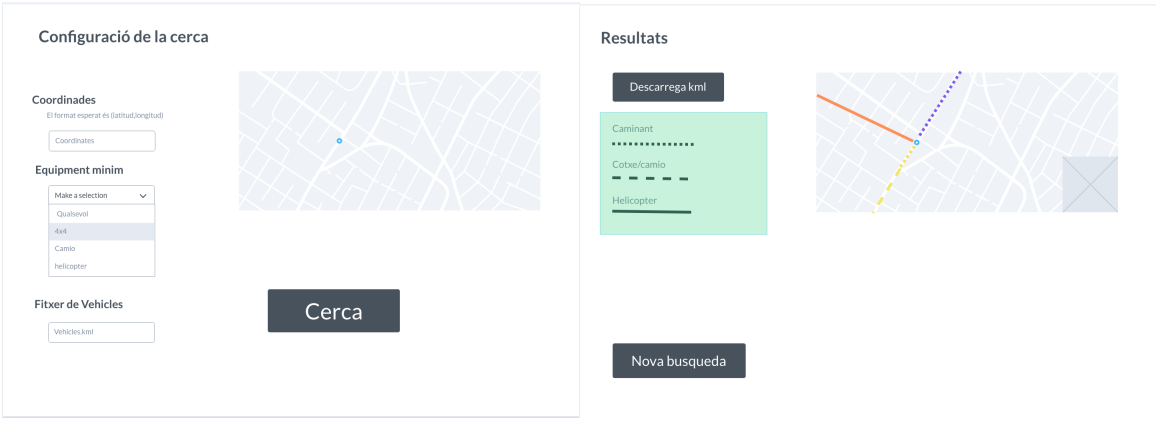
El primer dubte és quin tipus de software soluciona les necessitats dels bombers de la manera més útil possible. Com es tracta de un eina d'ajuda a la presa de decisions, amb un còmput elevat i la feina dels bombers no ens garantitza que tinguin un entorn adequat, es va decidir que es dividiria el sistema en un Frontend i un Backend, de manera que combinaríem un Frontend accessible des de qualsevol entorn, i la feina de còmput la traslladarà a un Backend que serveix una API-REST.

Tenint en compte l'extret en la reunió amb els bombers es va pensar que per que els bombers poguessin especificar la tipologia i ubicació del rescat, l'eina havia de tenir una entrada de coordenades, una per especificar els tipus de vehicles tenint en compte les necessitats del rescat i una per importar les posicions dels vehicles en el moment d'execució, a més d'un botó per poder fer la petició de l'obtenció de les rutes.

Un cop obtingudes les rutes, s'ha de redirigir a una pantalla on es mostren de forma clara, les rutes amb informació de les mateixes.

7.1.3 Sprint 2: Disseny de una primera interfície

En pos d'això es va fer un mockup, que detallés el comportament de l'aplicació a realitzar, introduint els requisits que van sorgir de la reunió i, es va optar perquè el mockup seguís un disseny de pàgina web, ja que ha de servir d'ajuda a la presa de decisions i que ha de ser accessible remotament:



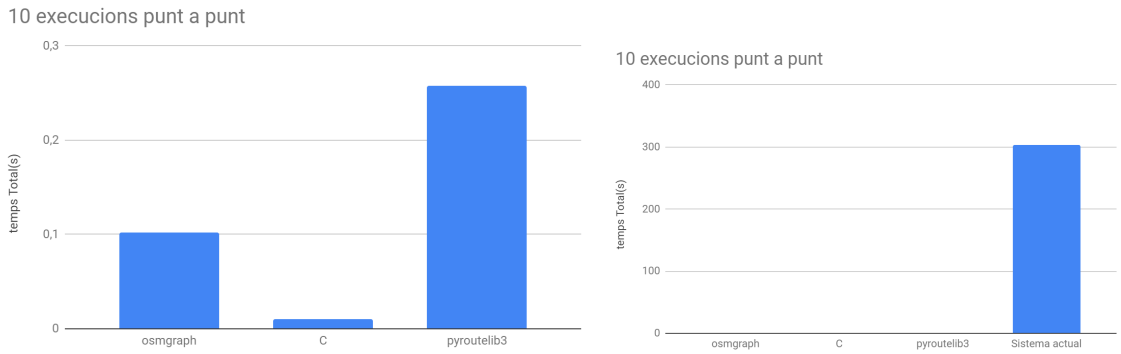
Il·lustració 35: Mock app de l'apilcació

7.1.4 Sprints 3 i 4: Estudi i primeres versions del algoritme d'exploració de grafs

A raó del objectiu 12,es va decidir d'usar python per les avantatges que ofereix el llenguatge, es a dir la seva sintaxi senzilla i la simplicitat de integrar-se amb noves eines. També es va tenir en compte la integració amb QGIS, ja que aquesta eina permet crear extensions en python.

Del sistema anterior es podia extreure que un dels grans impediments era el temps per a resoldre la ruta òptima, es van investigar tant el sistema anterior com d'altres llibreries tots en el llenguatge de programació python: pyroutelib3, osmgraph i un desenvolupat per mi mateix.

També es van comparar el rendiment amb nous programes que vaig desenvolupar amb el llenguatge de programació c per veure si el rendiment era realment superior en c, fins i tot en grafs tant petits com ho és el graf format pels camins de Montserrat. Això es va fer treballant amb la cartografia heretada del sistema anterior esmentat, en format osm. Ja que, dels estudiats, el format osm és el que està estructurat de manera que permet, més fàcilment, generar un graf dels estàndards cartogràfics estudiats.

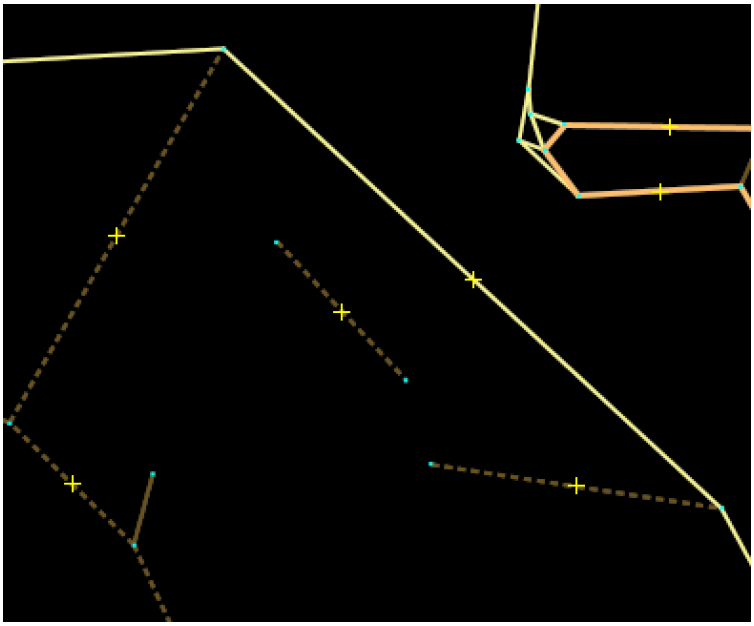


Il·lustració 36: resultats de l'execució per 10 camins diferents

Tots els programes de la gràfica es criden des de python, tot i que la solució en c empra per sota una llibreria que s'ha desenvolupat en aquest sprint. En base aquests resultats la decisió va ser que la part de resolució del graf es desenvoluparia en c ja que els rendiments de l'algoritme desenvolupat en c eren molt superiors als abans esmentats, superant els impediments en relació al temps per a resoldre la ruta òptima, tal i com es veu en les gràfiques anteriors.

7.1.5 Sprint 4 i 5: Validació del graf

En aquesta tasca és va fer servir JOSM per esbrinar si el mapa era representatiu dels camins de Montserrat, i es va desenvolupar un script nodesDesconectats, que donats tots els nodes explorava el graf i donava tot el conjunt de nodes que estaven connectats, després de descartar-los en decidia un altre i així fins a donar tots els subconjunts connexos. Això va servir per esbrinar que el mapa amb el que treballem, l'heretat del sistema anterior, no era totalment connexa.



Il·lustració 37: aresta inconnexa en JOSM

Fent una visualització sobre JOSM buscant per id els punts inconnexos es va descobrir que a part dels 3007 nodes del graf connex ni había 85 de inconexos, dividits en 14 grups, això es devia en 5 casos a la falta d'alguns senders, en el mapa que uneixen els representats per les arestes incones amb la resta, en la resta es va detectar que la connexió amb aquests punts es devia al limit que delimiten la nostre cartografia. No hi havia cap cas de un node sense aresta.

7.1.6 Sprint 5 i 6: Primera versió amb vehicle i a peu

Tenint una manera d'explorar el graf de manera efectiva el primer que es va fer es adaptar-la per al nostre problema. Així dons es va fer que calculi totes les rutes, sortint des de qualsevol punt amb el vehicle que se li demani per anar fins al punt de destí, essent capaç de decidir on deixar el vehicle per anar caminant, si s'escau, realitzant el que anomenaré un dijkstra en quatre fases.

Com la cartografia del sistema anterior ja comptava amb les dades de distancia, calculades tenint en compte els desnivells, només s'ha de calcular el temps que es tarda per a cada tram -que com hem dit, es corresponen als way de l'estàndard osm- Per a calcular aquest temps s'ha de tenir en compte dos supòsits:

a) Si es va en vehicle, el nostre límit de velocitat és només la velocitat màxima de la via pel vehicle en qüestió, així doncs el temps es calcula com a distància/velocitat.
b) Si no es va en vehicle, la velocitat ha de tenir en compte el desnivell, així doncs per calcular el temps es necessita les inclinacions -en realitat, la inclinació acumulada-

(rampa/distancia) i finalment el temps es calcula com: distancia/velocitat * (1 + inclinació positva)*(1+⅓ de la inclinació negativa)

7.1.6.1 Dijkstra en quatre fases

Els valors de la llista de precedències estan quadruplicats, es a dir tenim un *array* de la estructura:

```
struct PrecNodeDist
{
  int pred[4];
  double cost[4];
  int vehicle[4];
  double rampPos[4];
  double rampNeg[4];
  double dist[4];
};
```

Il·lustració 38: element de la llista de precedencies

Per cada un dels elements d'aquesta estructura la posició **0** corresponia a caminar, la **1** a anar amb cotxe, la **2** a anar amb BRP i la **3** a emprar un tot terreny. El valor **"pred"** es corresponia al node predecessor, "**cost**" al cost calculat per arribar del node actual al node destí, "**rampPos"** a la inclinació positiva de la aresta(way en osm), "**rampNeg**" a la inclinació negativa, "**dist**" feia referencia a la distancia de l'aresta i finalment "**vehicle"** al vehicle que s'emprava per navegar al pròxim node.

El primer que fèiem es recorre el graf emprant l'algoritme dijkstra a peu per tant, per tant un cop feta aquesta iteració teníem la llista de precedències, amb el recorregut a peu, en segon lloc per cada un dels vehicles, però per cada un dels vehicles també es tenia en compte si era millor deixar el vehicle i emprar la ruta a peu.

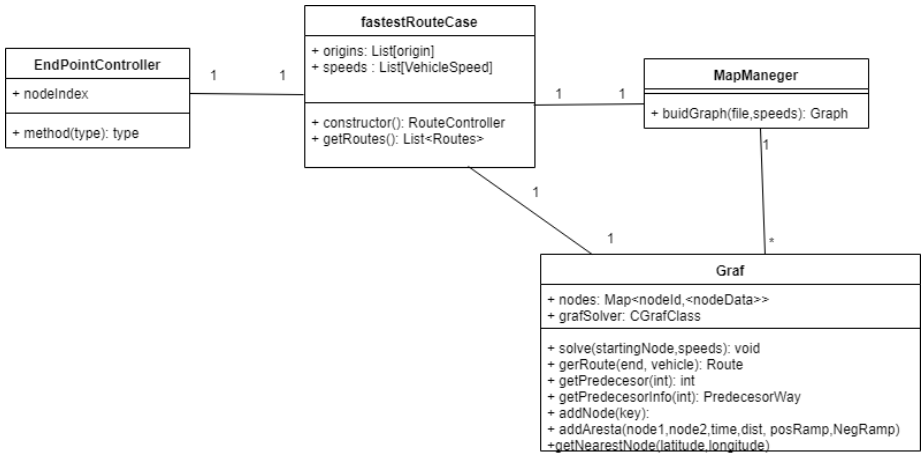
```
if (vehicle==1 && isInMinHeap(minHeap, v) && dist[u][0] != INT_MAX &&
                    pCrawl->walkingCost + dist[u][0] < pred[v].cost[1]
                    && pCrawl->walkingCost < pCrawl->CarCost)
{
  dist[v][1] = dist[u][0] + pCrawl->walkingCost;
  pred[v].pred[1] = u;
  pred[v].cost[1] = dist[u][0] +pCrawl->walkingCost;
  pred[v].vehicle[1]=pred[src].vehicle[0];
  pred[v].rampPos[1]=pCrawl->rampPos;
  pred[v].rampNeg[1]=pCrawl->rampNeg;
  pred[v].dist[1]=pCrawl->dist;

  // update distance value in min heap also
  decreaseKey(minHeap, v, dist[v][1]);
}
```

Il·lustració 39: Segent de codi on es comprova si es millor deixar el vehicle en aquest punt

En la condició anterior comprovem amb quin vehicle estem calculant, si v (node de on vindrem) es un dels nodes en el conjunt min heap (nodes on no s'ha comprovat la distancia mínima per aquesta iteració del dijkstra, recordem que se'n fa una per vehicle), comprovant que el nou cost sigui inferior al antic i que això sigui degut a que és millor anar caminant que en vehicle.

7.1.6.2 Implementació del Backend

Com s'ha explicat en la fase de disseny s'ha decidit crear un Backend en format API REST. Tot així en un futur es possible que s'empri la llibreria d'enrotament per generar una extenció de QGIS, així que el framework per generar les peticions ha de independent a la resta del sistema, tenint en compte això i els meus coneixements s'ha arribat a la conclusió d'emprar el framework flask per les peticions. També per maximitzar la independència s'ha decidit el següent esquema:



Il·lustració 40: Diagrama de classes de la primera versió del Backend

Com es pot veure en l'esquema anterior es va desenvolupar la classe fastestRouteCase. La idea va ser que aquesta classe tingués la lògica de identificar les rutes, des dels punts d'origen fins els de destí i el tornés en un format indexat.

La classe fastestRouteCase, al ser construïda, crida el MapManeger. El MapManeger llegeix el mapa i retorna la classe Graf, que és qui s'encarrega de interaccionar amb la llibreria de resolució de grafs en c esmentada en l'apartat anterior. Es demana a configuració la llista de diccionaris que corresponen a les velocitats per vehicle i tipus de camí, cada element del diccionari es correspon a un vehicle i per cada vehicle el diccionari conte a quina velocitat pot anar a cada via.

La classe Graf se'n encarrega de resoldre la interacció amb la llibreria en c. Un cop resolt el graf, per la llibreria de resolució de grafs, la classe Graf pot preguntar a la llibreria en c, quin ha de ser el següent punt a arribar si, des de qualsevol lloc de la cartografia, vols anar al punt de destí. Així doncs la classe Graf va demanant punt a punt per on ha de passar cadascú des dels diferents punts d'origen fins al destí. Un cop té les rutes les agrupa en un diccionari, les retorna a qui li demani en el nostre cas la classe fastestRouteCase.

Com dintre la cartografia que tenim fins el moment,(la de les immediacions del Parc de Montserrat), només hi ha, com a punt d'origen significatiu, el Parc de Bombers de Collbató, es van decidir 5 punts a l'atzar, com a punts de sortida dels vehicles. Aquest punts es va comprovar que estiguessin ben connectats, evitant que la base estigués enmig d'un sender només accessible a peu.

En aquesta versió es va crear una interfície feta amb Ionic per mostrar els resultats. I una API desenvolupada amb el framework de python flask s'encarregava de resoldre les peticions cridant el RouteHandler.

7.1.6.3 Problemes detectats:

Aquesta versió tenia el problema que permetia més d'un accés al punt de rescat o de destí (un per vehicle o equip a peu), és a dir que pot ser que pels diferents vehicles -camió de bombers, 4X4, ambulància- o accés a peu, la ruta no tingui cap punt de trobada entre els diferents mitjans; cosa que no es correspon amb les necessitats dels bombers. El que necessiten els bombers és que els vehicles i les persones es puguin combinar i, per tant, s'han de trobar en uns punts que són els que els bombers denominen punts de reunió.

Els camins que pintava aquesta versió no eren els autèntics, sinó una simplificació dels mateixos, perquè només pintava els extrems dels trams dels camins.

7.1.6.4 Avanços assolits

Els temps d'exploració del graf son molt prometedors
Tenim una interfície que mostrava les rutes de forma clara
Teníem en conte la distinció entre tipus de vies i vehicles

7.1.7 Sprint 7 i 8: Primera versió multimodal

Per poder forçar que les rutes calculades facin servir els punts de reunió tal i com s'ha mencionat en l'apartat anterior. es va decidir un nou procés de càlcul, que també és gestiona des del RouteHandler.

En primer lloc s'explora el graf amb un dijkstra amb la capacitat de canviar de vehicle en qualsevol punt, aquesta primera aproximació ens dona la ruta que, amb menys cost, ens aproxima combinant de tots els vehicles o equips de persones a peu. Aquesta versio del dijkstra sera el dijkstra Multimodal.

7.1.7.1 Dijkstra Multimodal

El dijkstra multimodal a diferencia del Dijkstra en quatre fases, no realitza diverses exploracions, si no una de sola. Així doncs qualsevol aresta del graf te 4 posibles pesos i l'algoritme els mirarà tots.

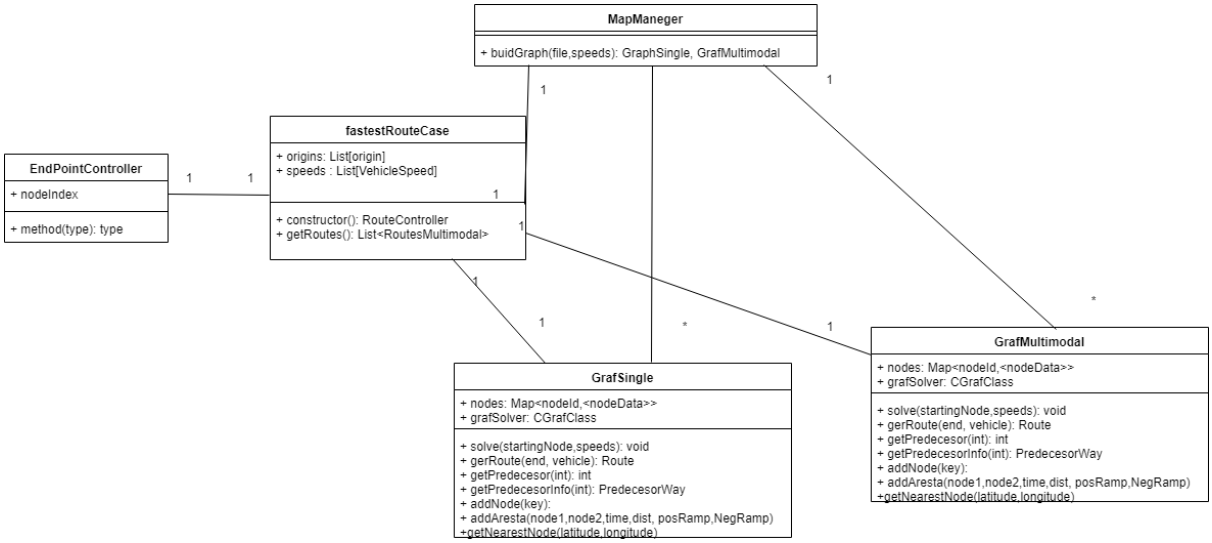
Aquesta primera solució a de ser una combinació de la utilització dels diferents vehicles

7.1.7.2 Implementació del Backend

Un cop calculades les rutes multimodals des de les bases fins el punt de la incidència, per cada ruta i vehicle s'extreu el punt màxim on pot accedir el vehicle, i es calcula la ruta de totes les bases fins aquell punt amb aquell tipus de vehicle. Un cop se sap el punt màxim on poden accedir tots els vehicles tenim el punt de trobada que ha d'identificar-se en el mapa. Aquests punts segons la definició dels bombers, han de coincidir o ser molt propers amb els punts de trobada que ells tenen ja definits.

S'ha demanat que a més del temps, es tornin distàncies, i altures, s'ha decidit que aquesta informació es busqui en el graf.

Així doncs el resultat de la API són tots els camins principals (els calculats pel dijkstra multimodal) i per cada camí com accedir-hi emprant el vehicle corresponent. Per representar això s'ha decidit una estructura json amb les rutes referenciades per node de sortida, com en la versió anterior, però ara, per cada subruta no nomes tindrem el seguit de punts que la comprenen si no també, la informació general de la ruta (cost i distància), i el camp subrutes amb les rutes dels vehicles fins a on poden arribar de la ruta.



Il·lustració 41: Diagrama de classes de la segona versió del Backend

7.1.7.3 Problemes detectats:

El graf que navega el dijkstra de vehicles en comptes de 4 grafs disposa de un graf amb quatre pesos diferents per aresta, resultant en que per les arestes que només existeixen per alguns vehicles, pels que no hauria d'existir te valors enormes per indicar assegurar que sigui la ultima opció possible. Així amb aquesta solució, si una aresta existeix per un vehicle, existeix per tots i si s'intenta fer viatjar un vehicle fins a un punt on en realitat no podria es pot aconseguir tot i que amb un cost descabellat.

Els camins que pinta aquest sistema segueixen sense ser els originals.

7.1.7.4 Avanços assolits

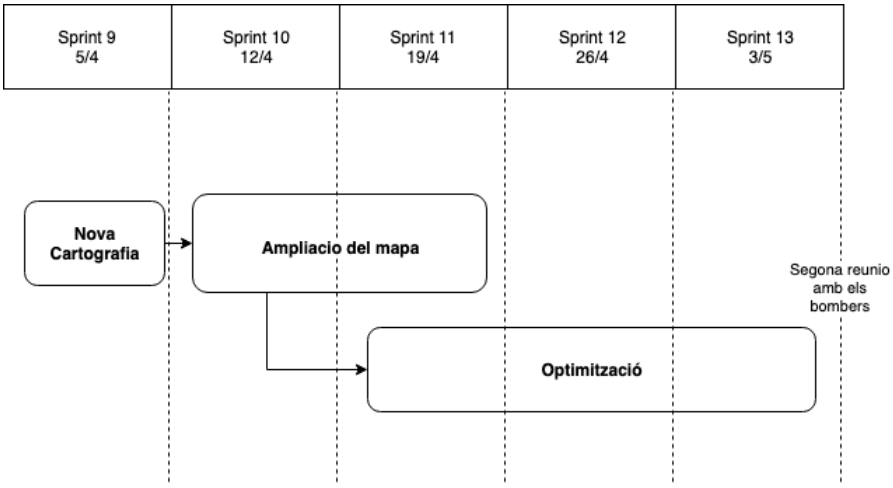
Els temps d'exploració del graf son molt prometedors i es te en compte la combinació de vehicles

Tenim una interfície que mostrava les rutes de forma clara

Teníem en conte la distinció entre tipus de vies i vehicles

7.2 Optimització de la cerca

En aquesta fase és genera la nova cartografia, i s'amplia el que ens porta a optimitzar el procés de cercar les rutes.



Il·lustració 42: Diagrama de Gant de la fase Optimització de la cerca

7.2.1 Sprint 9: Nova cartografia

Arribat aquest punt del projecte teníem la cartografia operativa d'emergència(COE), així que és decidí a incorporar amb el que ja es tenia. Sent que estava en format shape, es va seguir el procediment del sistema anterior per tal d'aconseguir el format desitjat. Aquest procediment comença passant les dades dels fitxers shape a una bd postgres, el resultat són tres taules:

- **COE_MONTSERRAT_LINEES** el qual conté els diferents camins especificats en la cartografia operativa dels bombers i serà el que convertirem en OSM emprant els scripts del sistema anterior
- **COE_MONTSERRAT_PUNTS** que conté els punts d'interès que els bombers tenen senyalitzats. Aquests punts haurien d'aparèixer en la visualització.
- **COE_MONTSERRAT_POLIGONS:** límits territorials de les àrees de bombers, no apareixen en aquest projecte

Acte seguit s'utilitzen els scripts del sistema anterior per passar les dades de la taula COE_MONTSERRAT_LINEES a un fitxer osm amb els tags de distàncies, rampes i altituds que requerim.

Un cop es tenen els camins a la base de dades, es van executar els scripts del Sistema anterior per passar les dades a un arxiu Osm i simplificar els trams, desant els valors de distància i desnivell reals.

7.2.1.1 Problemes detectats:

Al realitzar el procés esmentat s'ha trobat que només s'han transmès els camins de tipus senders de manera que no tenim cap dada de vehicles de bombers, però s'han vist quina estructura tenen les dades i que la columna name definiria el tipus de marcatge (Com només tenim senders, disposem tant sols del color de la retolació).

També s'ha detectat que el script utilitzat recull les dades dels camins d'un fitxer de configuració que s'estructura de manera que cada taula és d'una tipologia individual, el que no coincideix amb l'estructura que tenen les dades, aquesta situació s'haurà de modificar en un futur per a poder situar les rutes com requerim.
S'ha detectat una evident pèrdua de punts en la generació del mapa, que fa perdre precisió a l'algoritme

7.2.2 Sprint 10 i 11: Ampliació del mapa

Per poder mostrar que disposàvem d'un algoritme potent i validar-lo es va decidir concertar una segona reunió amb els bombers i afegir punts d'interès reals com a punts de sortida. Com encara no és tenia un fitxer de vehicles i l'objectiu era demostrar que era un sistema funcional, es va decidir incorporar els parcs de bombers. Els parcs decidits foren els de Terrassa, Igualada i Manresa, que son els que es van trobar més propers fent una cerca en google. Com que tots aquests parcs queden fora de l'entorn de Montserrat es va decidir ampliar el mapa. Es va agafar l'àrea que comprèn des de Solsona a Barcelona. Aquestes dades es van extreure de OSM utilitzant al API Overpass, d'aquesta informació es va fer un cribratge excloent-hi la informació no necessària mitjançant l'eina OSMOSIS. Concretament es van extreure tots els punts que no fossin de camins i tots els senders i pistes forestals que ja estan especificats amb mes precisió i detall a la cartografia de bombers. Un cop es tingueren les dades filtrades es va fusionar amb el osm extret de les dades dels bombers emprant JOSM tal i com s'explica a l'apartat de eines utilitzades. Amb aquesta mateixa eina es van estudiar i resoldre els errors de nodes repetits, ways creuats, ways pròxims a altres ways però inconnexos amb el procediment detallat a l'apartat de detecció i correcció d'errors.

7.2.2.1 Problemes detectats:

Un cop es va tenir l'arxiu osm, passant els scripts per afegir la informació d'altures i el de simplificar el graf es va detectar que amb un nombre tan gran de punts el procés tarda de l'ordre de 4 dies, tres per l'script de OSM2NET.py i un per el NET2GRAPH. Aquest gran retard es causat per l'abús que fan aquests scripts en memòria, fet produït per que les cerques i

l'emmagatzematge sobre els nodes recorreguts no es fan de manera indexada i que també perquè es fa en un llenguatge interpretat (python).

7.2.2.2 Avanços assolits:

Finalment fem servir les dades que proven del COE

7.2.3 Sprints 11, 12 i 13: Optimització

Amb la nova cartografia es va decidir utilitzar el nostre sistema, però els resultats de la API tardaven de l'ordre de minut i 20s, que es molt per sobre del que es pot considerar un temps acceptable.

Per tal de resoldre això és va analitzar la classe RouteHandler mitjançant la llibreria de python time i així esbrinar quines parts del algoritme ocupaven més temps per processar. El resultat va ser que la part que tardava més era la encarregada d'extreure les dades de distàncies i altures de cada ruta que es feia en temps d'obtenció dels resultats de la exploració.
Això va resultar una mica sorprenent. El cost d'extreure les dades de les rutes havia de ser lineal, mentre que el cost d'exploració era el que havia de ser mes costos d'ordre quadràtic.

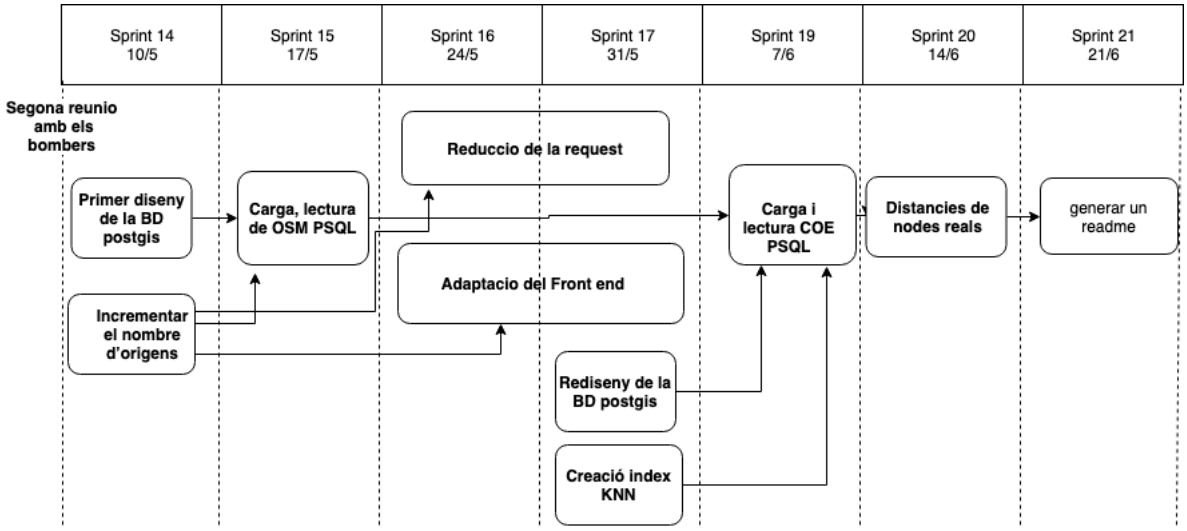
Així dons tenint en compte la llei d'Amdahl. Ens vam centrar en minimitzar l'impacte d'aquest procés. Es va decidir que en comptes de només l'identificador del node destí, la llista de precedències també guardés un punter a l'aresta utilitzada, d'on extreure distancia, costs, rampa positiva i negativa acumulades. Fent això enviàvem tenir que fer una cerca de l'aresta en el graf. De manera que va resultar menys significativa, es va limitar l'exploració del dijkstra de vehicles. Fent que en comptes del graf sencer, només és recorregués fins ha trobar tots els punts que fem servir com a punts de sortida. El resultat va ser que la part de API tardava entre dos i tres segons.

7.2.3.1 Objectius assolits

Els temps d'exploració del graf son molt prometedors passem de 80-90s segons a 2-3s
Tenim una interfície que mostrava les rutes de forma clara
Tenim en conte la distinció entre tipus de vies i vehicles
La solució a de buscar la combinació de vehicles
Hem augmentat l'àrea de la cartografia, però encara no estem al ordre de tot Catalunya.

7.3 Optimització de precarga

En l'ultima fase s'ha augmentat el nombre d'orígens incorporant el fitxer de vehicles, s'ha incrementat el mapa a tot Catalunya en concordança i finalment s'ha optimitzat tot el proces de precarga



Il·lustració 43: Diagrama de Gant de la fase optimització de precarga

7.3.1 Segona reunió amb els bombers

Es va realitzar una segona reunió amb els bombers se'ls hi va presentar l'avanç realitzat i van veure que era una eina funcional i pràctica, sobretot pel rendiment i la capacitat de veure sobre mapa també es va considerar una bona aportació el afegir punts que fossin parcs, però van sol·licitar que es realitza un redissenya per donar-li mes claredat, com la necessitat d'incorporar un input textual per les coordenades (ja que moltes vegades el que coneixen es la coordenada del lloc), un input per el fitxer de vehicles(ja que es va pensar en què importes des de el mòbil) i un extra per limitar el desnivell a peu.

Es va dir que el conjunt de parcs de bombers des d'on s'estudiava l'accés serien els de (Martorell, Collbató, Igualada, Castellfollit del boix, Manresa i Viladecavalls) ja que des de qualsevol altre parc s'hauria de passar per una d'aquestes poblacions per arribar a Montserrat.

També es va donar un exemple de fitxer de vehicles. Que es farà servir pel desenvolupament. El fitxer de vehicles es un kml on es mostren la posició i tipus dels diferents vehicles i està sincronitzat en temps real via Dropbox.

Il·lustració 44:Visualització del fitxer de vehicles

7.3.1.1 Problemes detectats:

Provant l'aplicació amb els bombers es va descobrir que hi havia punts on es podria anar en vehicle però el sistema no ho tenia en conte. A arrel d'aquest error trobat es va revisar la cartografia, descobrint que segons la cartografia dels bombers allò era un sender i per tant l'error estava en els inputs d'entrada.

També en un dels punts es va trobar l'error mencionat en l'apartat del dijkstra únic on es podia forçar un vehicle a anar per un entorn que li era impossible arribar a causa de que s'havia deixat un segment osm de camí transitable pel vehicle en un punt on només s'hi pot accedir a peu.

Els camins que pinta aquest sistema segueixen sense ser els originals.

7.3.2 Sprint 14: Incrementar el nombre d'orígens

Es va afegir a la arquitectura la classe OriginsHandeler aquesta classe s'encarrega de demanar a les diferents fons la informació sobre els orígens i tornar-la com a json.

S’inserí les bases com a json que el OriginsHandeler recull. També és crea el kmlHandler que gestiona el fitxer kml de vehicles i el retorna com a json cap al OriginsHandeler

Sumant les bases i els punts extrets de vehicles es troben 60 punts i el Backend tarda en tornar les dades entre 50 i 60 segons. Tenint en compte que en un rescat sigui quin sigui l’origen els punts de destí dels vehicles i els de trobada son reduïts, es va decidir agrupar els càlculs de vehicles i així evitar repetir cerques, amb aquests canvis el temps de crida va tornar a baixar a uns 10-15 segons però el temps de càlcul es trobava per entre els dos-tres segons. Per tant la resta del temps que emprava estava relacionat en operacions del framework FLASK.

7.3.2.1 Problemes detectats:

- Es va detectar que per cada crida a la API tornaven uns 120MB, s’havia de reduir, perquè això podia ser un dels factors del retard i a més parlem que el Frontend pot ser en un mòbil on la quantitat de dades s’ha de tenir en compte.
- Els camins i punts que pinta aquest sistema segueixen sense ser els originals.
- Es va detectar que alguns vehicles no podien arribar al massís per que faltaven camins al mapa.

7.3.2.2 Objectius assolits

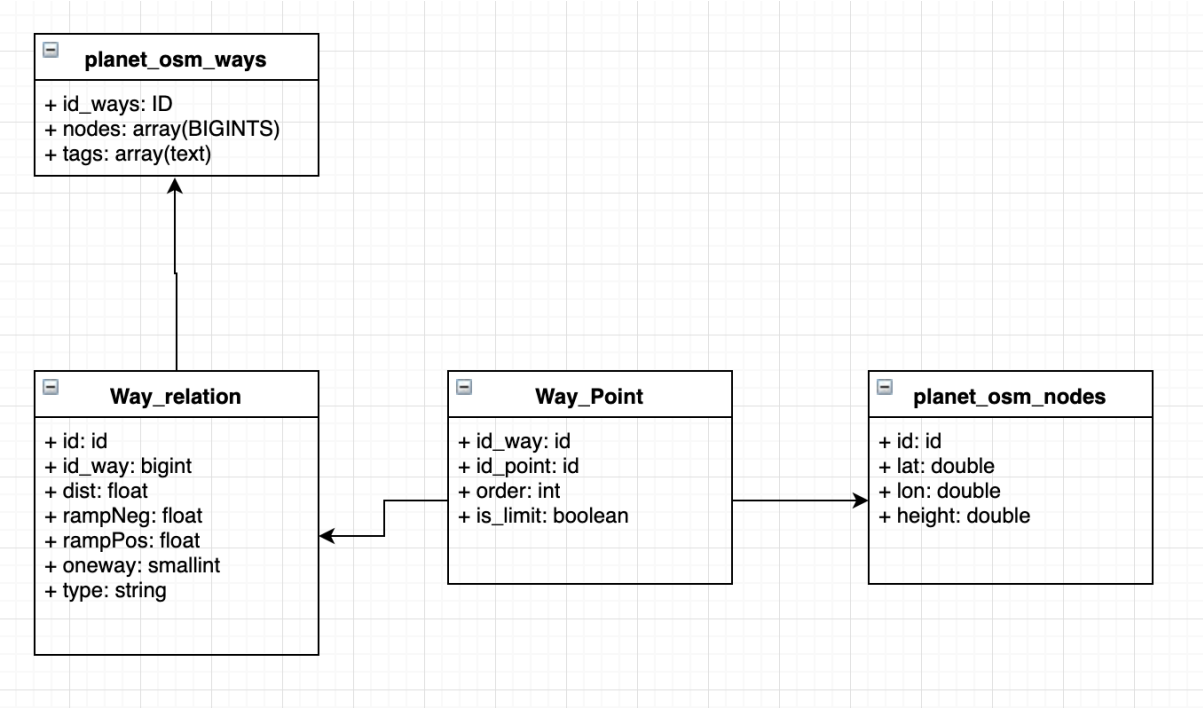
- Els temps d'exploració del graf son molt prometedors passem de 70 segons a 10-15(Objectiu 1)
- Hem augmentat l'àrea de la cartografia, però encara no estem al ordre de tot catalunya que seria l’objectiu 8
- El sistema permet la configuració del posicionament dels recursos de bombers (Objectiu 6)

7.3.3 Sprint 14: Primer disseny de la BD postgis

Com es va detectar problemes en les dades de la cartografia, que l’algoritme feia servir tenia segments que no estaven ben especificats, s'havia d’arreglar, i per fer això, s'havia de retocar el osm original, però això implicaria tornar a passar els scripts simplificar els trams, desant els valors de distància i desnivell reals. Que com s’ha detectat abans, suposaria perdre prop d’una setmana per fer un procés que comportava la pèrdua de precisió

Aquest fet obligava a implementar una solució alternativa i es decidí que la opció més òptima era realitzar tot el procés des de la mateixa base de dades(POSTGIS) on s’havien dipositat

les dades dels bombers per tal de construir el fitxer osm. Això no només va accelerar el preprocés de les dades cartogràfiques, si no que va donar-li a la plataforma una manera de guardar i referenciar les dades del graf amb les del mapa real, fent possible després dibuixar els camins tal com són. I també permetria afegir la funcionalitat d'extreure la informació de l'especificació dels bombers sobre camins i pistes.
Per poder fer aquest procés amb menys temps i eliminar la pèrdua de precisió es va decidir el següent model



Il·lustració 45: Diagrama de taules de la primera estrucutura de la base de dades

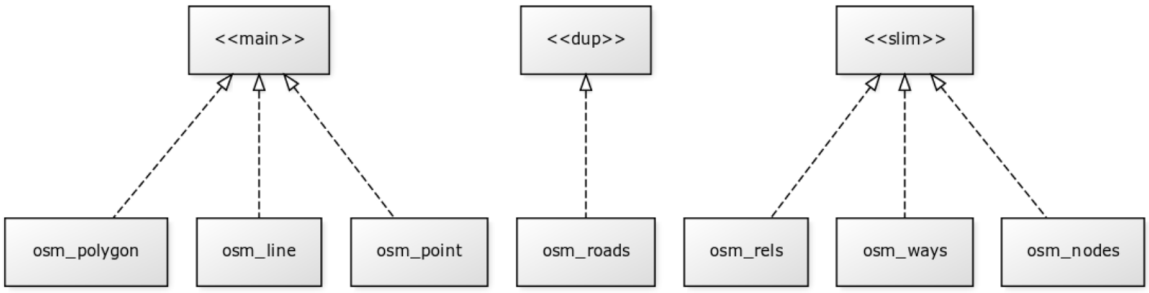
7.3.4 Sprint 15: Carga, lectura de OSM PSQL

En el sistema anterior, com les dades de bombers venen en format d'agulles que com s'ha dit al l'estat del art li manquen relacions de node a aresta com les que necessitàvem per navegar al graf. Hem de generar igualment el osm i revisar que aquestes dades es van crear correctament de forma manual. Aquestes dades s'han de lligar amb les de la API Overpass. El problema es que amb la quantitat de dades, l'eina que es va fer servir pel procés manual de unir les dades dels bombers amb les del osm (JOSM) es sobrecarregava així que per aquesta iteració es va decidir prescindir de les dades COE i fer servir les del model OSM el procés a seguir va ser:

Es va descarregar fent servir la API Overpass la àrea decidida del mapa de catalunya

Seguidament passar les dades a osm2psql i l'opció **slim** el que crearà les taules que necessitava el nostre algoritme d'enrutament.

El resultat va ser el següent:

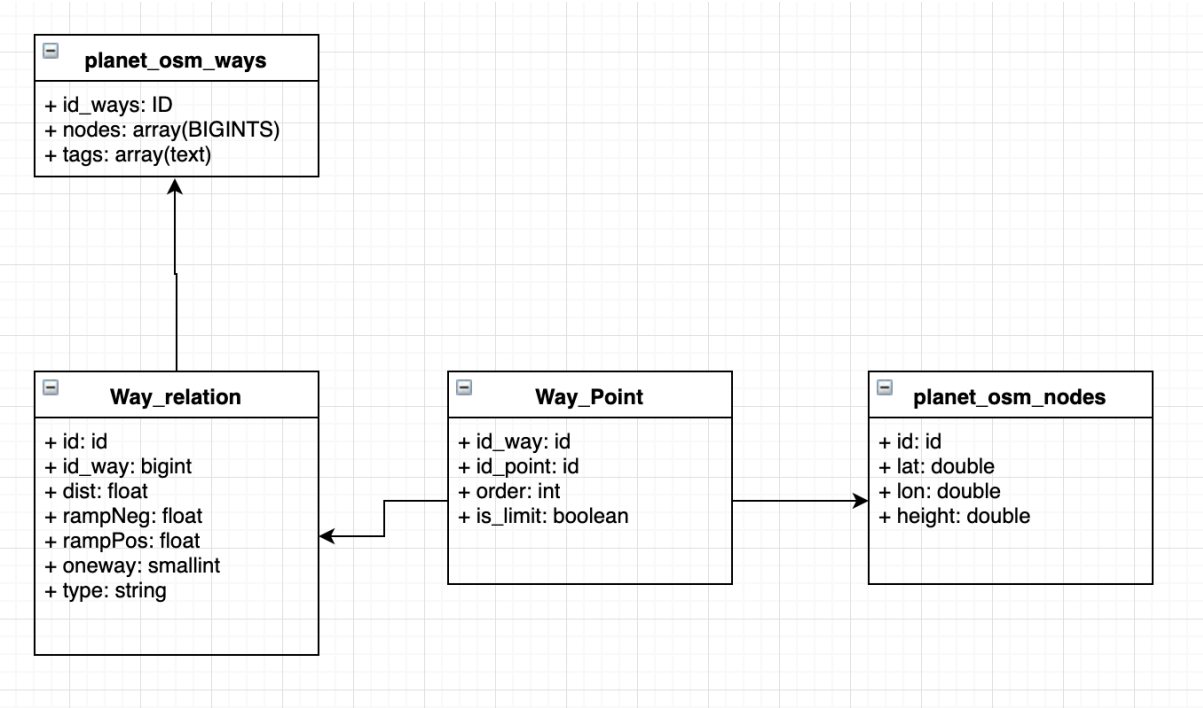


Il·lustració 46: Esquema de l'estructura de dades generada per osm2psql

D'aquí les taules que interessen més eren osm_ways i osm_nodes.

Mitjançant la classe manageDBData es van gestionar els canvis adequats per que el graf pugues treballar amb aquestes dades.

En primer lloc es cridava la funció sql **update_node_way** que canviava l'estructura que dona **osm2psql**, afegint les taules way_relation i way_point per relacionar les taules planet_osm_ways amb planet_osm_nodes i poder realitzar els talls amb la funció sql **realitza_talls** tal com feien els scripts del sistema actual OSM2NET i NET2GRAPH, però mancaven les distàncies, el desnivell i les restriccions de la via, que es calculaven amb la funció sql **calculaDistances** després de que **manageDBData** poses les altures als diferents nodes a partir de les dades extretes del fitxer **raster montserrat-dem.vrt.**



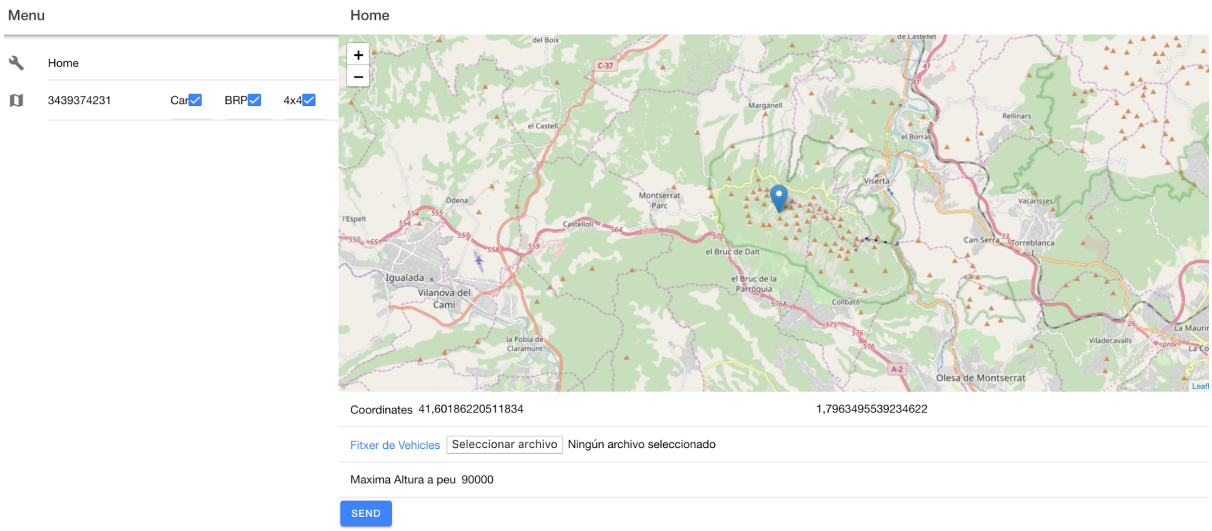
Il·lustració 47: Diagrama de l'estuctura de la primera versio de bd

Ara que ja tenim totes les dades a bd, aconseguíem evitar la lectura del fitxer OSM. Que obligava a posar tot el fitxer en memòria. També gracies a això es pot implementar un sistema per modificar la cartografia. També ens permetrà emprar les instruccions CRUD per modificar la cartografia, el que en un futur pot derivar en un sistema de modificació.

7.3.5 Sprint 16 i 17: reducció de la request i Adaptació del Front end

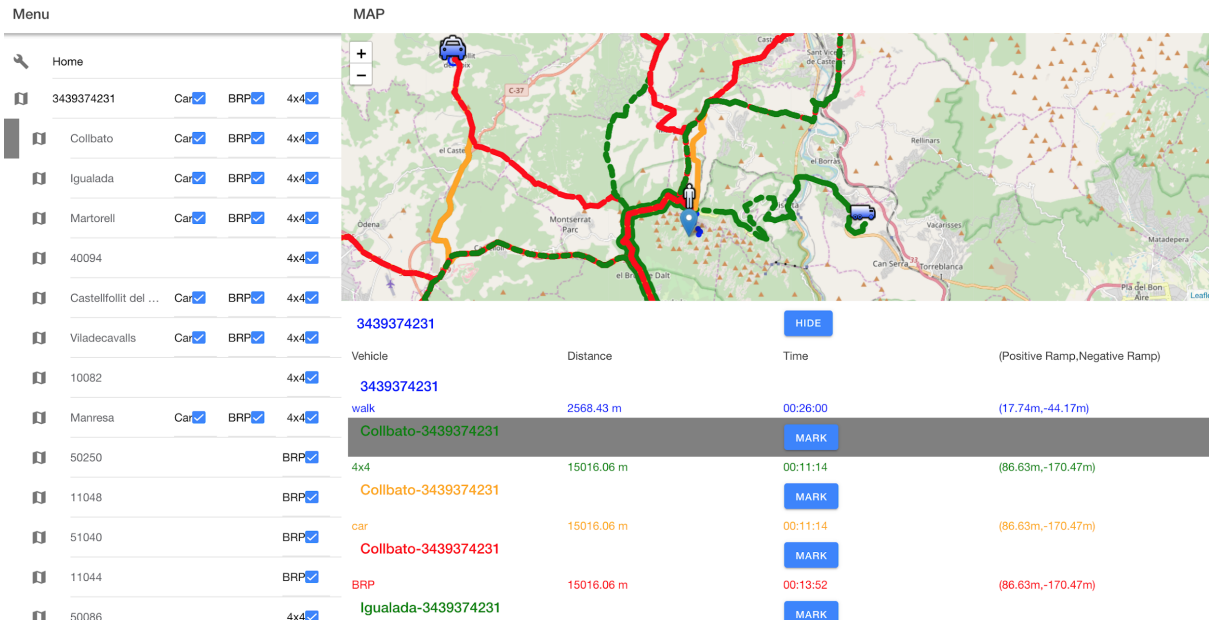
Aprofitant que es canviava el Frontend, per complir les especificacions dels bombers, també es va canviar l'estructura de les dades a tornar. Així doncs les subrutes no estarien dins de les rutes a les que van dirigides, si no que es retornaran les rutes fins l'objectiu per una banda i les rutes fins el punt de trobada per un altre. Solucionant així el problema de l'excés en el retorn de dades.

Aquest nou Frontend es compon de dos vistes, una amb el formulari per especificar les coordenades, afegir el fitxer de vehicles o seleccionar una diferencia d'altura màxima per limitar les rutes possibles. Per les coordenades a part de escrivint també es podien seleccionar clicant a un mapa que es mostra en aquesta vista.



Il·lustració 48: Captura de la pantalla de seleccio del punt de rescat

Un cop feta la cerca al menu apareixia el o els nodes que l'algoritme decididia com a possible punt de Trobada. Si el clicavem ens apareixia la segona vista, on al menu hi apreixien tots els punts de sortida, amb opció de triar si es volia que es mostresin al mapa. A sota del mapa hi apareixien les dades de temps, distancia i diferencia d'alçada per realitzar la Ruta. Tambe hi ha la opcio de resaltar la ruta amb la opcio mark.



Il·lustració 49: Captura de la pantalla de visualitzacio de les rutes per un punt de trobada

També seguint les especificacions dels bombers es va afegir un endpoint per carregar arxius des de front end.

7.3.5.1 Problemes detectats:

Els camins i punts que pinta aquest sistema segueixen sense ser els originals.

La càrrega del fitxer de vehicles depenia de l'usuari tal i com havien demanat els bombers, però automatitzant la sincronització es podria reduir l'error humà i simplificar el flux. Ja que si la maquina obtingues el fitxer de manera automàtica, l'usuari no hauria de pensar en quin es l'últim fitxer, ni tampoc hi hauria el perill que s'equivoqués.

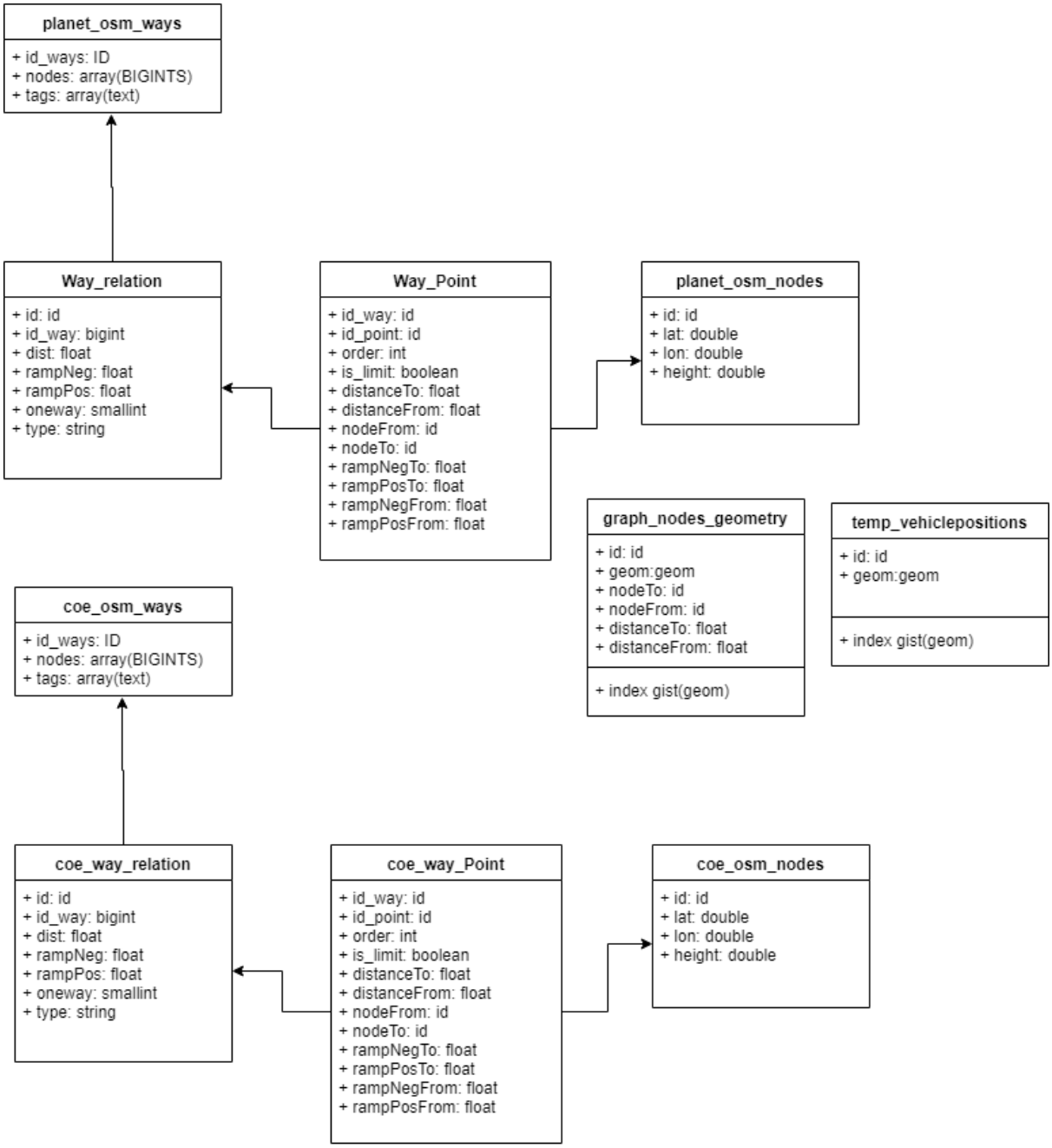
No s'havia afegit la cartografia de bombers de manera que no es tenien les seves dades. Nomes es contava amb les dades de OSM

7.3.5.2 Objectius assolits

La solució es fa extensible a tot catalunya(Objectiu 7)
Estem a l'ordre de 20s

7.3.6 Sprint 17: Disseny de la BD postgis

A fi d'acceptar no nomes els nodes del graf si no també els nodes del graf es va efectuar un redisseny de la bd que tingues en conte les distancies dels punts intermitjos fins els punts limítrofs del seu way e inclogués les dades de OSM, aquesta estructura es la que mostrarem a continuació



Il·lustració 50: Diagrama de l'estructura final de la base de dades

7.3.7 Sprint 17: Creació Join kNN

En aquest punt el procés de carga era molt lent, en concret ubicar els vehicles es tarava de l'ordre de 2 minuts, teníem 220 vehicles i consultàvem del ordre de $3*10^5$ nodes per saber quin era el punt en el graf.

Per resoldre això es va decidir de fer una Join entre els vehicles existents i els punts del graf, la Join havia d'unir els 220 vehicles amb els seus nodes del graf mes propers, això es va fer

fent servir una Join KNN optimitzada amb un índex GIS en les dos taules, emprant les eines que ens dona POSTGIS
La ubicació dels vehicles dins el graf passa a tardar de l'ordre de **2 minuts** a **0,19s**

7.3.7.1 Objectius assolits

Gracies a aquest estalvi de temps, podem anar actualitzant cada 5 minuts la ubicació dels vehicles de bombers resolent el *input* de les dades dels vehicles

7.3.8 Sprints 19: Carga i lectura COE PSQL

Seguidament s'extreu el osm de la cartografia de bombers amb el nou script psql2osm, aquest script busca a les taules especificades en el fitxer de configuració COETables, on també s'extraurà la informació de quin valor tindrà el camp **emergency_tag** que marcarà com esta indicat el camí a la cartografia COE de bombers. El valor del camp **emergency_tag** dependrà del camp name i folder del COE

Després de la validació i edició permitent emprant **JOSM** es va aplicar **osm2psql** amb l'opció **--prefix** i **--slim,** aquesta opció ens permet posar les dades dels bombers respectant les dades anteriors de osm i seguir el mateix procediment que l'anterior per posar les dades al graf

7.3.8.1 Objectius assolits

Gracies a aquest estalvi de temps, resolem i simplifiquem les dades COE, sense dependre de la simplificació del arxiu OSM que venia unit a una pèrdua de punts i per tant de precisió. Ara les dades de ambdós grafs es fan de forma diferenciada però des de postgres

7.3.9 Sprint 20: Distancies de nodes reals

En aquest punt del projecte només es treballava amb la simplificació de la cartografia, i tant sols es tenien en compte aquests punts, fent que el punt d'origen i el de destí forcin una mica imprecisos. El disseny de la base de dades ja tenia en conte això Per resoldre-ho la crida al sqlHandeler nearest node, no tindrà en compte tant sols els punts del graf, si no tots els punts que corresponen a les vies especificades al mapa. Així que en comptes de fer servir la vista **final_nodes** per omplir les dades de **graph_nodes_geometry** s'emprarà les dades de la vista **all_nodes**

Com s'ha vist en apartats anteriors graph_nodes_geometry ja te les dades per conèixer el cost d'arribar fins el punt.

7.3.9.1 Objectius assolits

Ara podem lligar els punts reals amb els punts del mapa. Amb el que resolem la pèrdua de precisió que venia acompanyada de nomes relacionar els punts amb els que son representatius del graf.

7.3.10 Sprint 21: Generar un Readme

Per poder possibilitar l'entrega del projecte, s'ha desenvolupat un Read.me amb la especificació de tot el desplegament del projecte, per tal que pugui ser desplegat i funcional.

8 Conclusions

En aquest capítol és descriuran les contribucions aportades en aquest projecte, els objectius realitzats, així com una valoració personal del treball realitzat. A mes es plantejaran les possibles millores o ampliacions que podrien sortir a partir del desenvolupat en aquest projecte.

8.1 Contribucions

La realització d'aquest projecte aporta als bombers de la Generalitat de Catalunya d'un sistema de suport per a la presa de decisions en els rescats de muntanya.

Amb aquest sistema trèiem nomes part de la responsabilitat de la decisió del camí a les persones. Ja que sent un sistema de suport a les decisions, la decisió final segueix sent dels bombers. Però ara no només es basarà en l'experiència dels bombers més experimentats, sinó també en mesures preses per un sistema automàtic.

Cal senyalar que aquest sistema automàtic no es basa en cartografies d'us quotidià, com la majoria de sistemes del mercat, si no en la cartografia operativa d'emergències(COE) que fan servir els bombers per el rescat i en els desnivells calculat per l'Institut Cartogràfic i Geològic de Catalunya. El que realment li dona al nou sistema el coneixement dels bombers més experimentats

8.2 Assoliment dels objectius

L'Objectiu principal d'aquest projecte era el de construir i dissenyar una eina de suport a la presa de decisions a la realització de rescats de muntanya.

Per tal d'assolir aquest objectiu s'han complert els subjectius especificats en l'apartat 8.4:

1. S'ha creat un sistema que dóna diverses solucions optimitzades pel rescat de muntanya a Montserrat. Això s'ha fet tenint en compte les restriccions donades pels bombers de la generalitat
2. El sistema té en conte desplaçaments en camió, en 4x4 i a peu.
3. El sistema arriba a traçar rutes per a tot Catalunya. Per donar resultats d'altres parcs com a resultat nomes cal afegir-ne la cartografia COE
4. El sistema permet explorar l'arbre de possibilitats amb el mínim cost en temps d'accés, recursos i desnivell acumulat a peu.

5. El sistema mostra de forma clara e intuïtiva els resultats.
6. El sistema dona eines per facilitar la decisió entre futures rutes.
7. El desenvolupament del sistema ha tingut en tot moment en compte ser escalat en un futur pròxim a nivell de funcionalitats.

8.3 Conclusions personals

Realitzar un projecte tant lligat al mon GIS ha estat tant un repte com una motivació. Tot i que la planificació era ajustada, el treball continu i l'esforç es van veure recompensats setmana rere setmana amb la progressió del projecte i la realització dels objectius proposats.

He ampliat molt el meu coneixement d'eines GIS, dels sistemes de coordenades i de georeferenciació que existeixen. També he après tot ha emprar tot un conjunt d'eines que segur que em seran d'utilitat en un futur pròxim.

També m'ha sorprès la potencia de dijkstra. Des de el principi del projecte pensava que al complicar la cartografia acabaríem derivant a un algoritme més complex.

8.4 Treball futur

Tot i que els objectius s'hagin assolit correctament, el projecte presenta alguns punts a partir dels quals es pot estendre o millorar el mòdul desenvolupat.

Cara a un futur escalament del projecte a tot catalunya requeriria de mes parcs amb més cartografies COE. De manera que el procediment seria el mateix que l'actual, però tenint en compte altres parcs.

Una altre possible millora seria passar més lògica de la lectura d'informació de les rutes a la llibreria en c. La complexitat de l'exploració és molt més alta que la de capturar la ruta però en el graf unimodal el temps acaba sent fins i tot superior. Això es per que la part de lectura de les rutes es fa en Python i nomes és demana el predecessor de cada node a la llibreria en c. S'hauria de generar una estructura en c amb tota la ruta i que es gestiones des de Python.

També es podria considerar l'accés d'ambulàncies i/o helicòpters per l'evacuació del accidentat. Actualment és calcula fins a on pot arribar una ambulància. Però les rutes de tornada no han entrat en el marc d'aquest projecte. Ni els desplaçaments en helicòpter.

A més és pot decidir ampliar el sistema per tenir en compte la capacitat de carrega.

És podria també afegir la possibilitat de fer Geofencing d'àrees excloent camins que passin per certs espais que poden estar relacionats amb la incidència i poden ser inestables.

Finalment també seria interesant tenir en compte la meteorologia ja que pot afectar en el rescat.

9 Bibliografía

1. wikipedia. *wikipedia.* [En línea] https://ca.wikipedia.org/wiki/Grup_de_Recolzament_d%27Actuacions_Especials.

2. wikipedia. *wikipedia.* [En línea] https://ca.wikipedia.org/wiki/Grup_de_Recolzament_d%27Actuacions_Especials.

3. Interior salvaments al medi natural. *Interior Gencat.* [En línea] http://interior.gencat.cat/ca/arees_dactuacio/bombers/seguretat_a_la_muntanya/salvaments_al_medi_natural/.

4. Cartografia operativa de la muntanya de monsterrat. *Interior.* [En línea] http://interior.gencat.cat/ca/arees_dactuacio/bombers/seguretat_a_la_muntanya/muntanya_de_montserrat/cartografia-operativa-demergencies-de-la-muntanya-de-montserrat/.

5. Grup de treball de seguretat a la muntanya de Montserrat. *Interior.* [En línea] http://interior.gencat.cat/ca/arees_dactuacio/bombers/seguretat_a_la_muntanya/muntanya_de_montserrat/grup_de_treball/.

6. generalitat, Bombers de la. Interior manual inventari camins Bombers. *Interior.* [En línea] http://interior.gencat.cat/web/.content/home/030_arees_dactuacio/bombers/foc_forestal/publicacions_tecniques_i_normativa/guies_tecniques/prevencio/manual_inventaris_COE/2017_MANUAL_INVENTARI_CAMINS_Bombers.pdf.

7. bombers, Manual inventari de camins de. Interior manual inventari de camins de bombers. *Inerior.* [En línea] http://interior.gencat.cat/web/.content/home/030_arees_dactuacio/bombers/foc_forestal/publicacions_tecniques_i_normativa/guies_tecniques/prevencio/manual_inventaris_COE/2017_MANUAL_INVENTARI_CAMINS_Bombers.pdf.

8. Wikipedia Agil. *Wikipedia Agil.* [En línea] https://en.wikipedia.org/wiki/Agile_software_development.

9. Cockburn, Alistair. agilealliance. *agilealliance.* [En línea] https://www.agilealliance.org/agile101/.

10. manifesto, Signants del agile. Agile manifesto principles. *Agile manifesto principles.* [En línea] https://agilemanifesto.org/iso/ca/principles.html.

11. Agilealience. *Agilealience.* [En línea] https://www.agilealliance.org.

12. Scrum, the scrum guide. *The scrum guide.* [En línea] https://www.scrum.org/resources/scrum-guide?gclid=Cj0KCQjwjMfoBRDDARIsAMUjNZo8mR_YqGXuF23CPis5eQafNIYaa1-Ecvtle6_s4_5167eXKyyG-fIaAuJyEALw_wcB.

13. AgileforAll. [En línea] https://agileforall.com.

14. Manifesto Signants del Agil. agilemanifesto. *agilemanifesto.* [En línea] https://agilemanifesto.org/iso/ca/manifesto.html.
15. Wikipedia transporte multimodal. *Wikipedia transporte multimodal.* [En línea] https://en.wikipedia.org/wiki/Multimodal_transport.
16. wikipedia intermodal transport. *wikipedia intermodal transport.* [En línea] https://en.wikipedia.org/wiki/Intermodal_freight_transport.
17. *Source: UNDP Disaster Management Training Programme, Logistics Module (1st ed.), p.18.*
18. Burcu Balcik, Benita M. Beamon, Karen R. Smilowitz. *Last Mile Distribution in Humanitarian Relief.* s.l. : J. Intellig. Transport. Systems 2008.
19. opentripplanner. *opentripplanner.* [En línea] https://www.opentripplanner.org/.
20. optaplanner. *optaplanner.* [En línea] https://www.optaplanner.org/.
21. citymapper. *citymapper.* [En línea] https://citymapper.com.
22. openstreetmap wiki, PyrouteLib. *openstreetmap wiki, PyrouteLib.* [En línea] https://wiki.openstreetmap.org/wiki/PyrouteLib.
23. github, osmgraph. *github, osmgraph.* [En línea] https://github.com/Mapkin/osmgraph.
24. brouter. *brouter.* [En línea] http://brouter.de/brouter/.
25. Felner, Ariel. *Dijkstra's Algorithm versus Uniform Cost Search or a Case Against Dijkstra's Algorithm.*
26. github Contraction-Hierarchies. *github Contraction-Hierarchies.* [En línea] https://github.com/cc-routing/routing/wiki/Contraction-Hierarchies.
27. wikipedia shortest path algorithm. *wikipedia shortest path algorithm.* [En línea] https://en.wikipedia.org/wiki/Shortest_path_problem.
28. onicframework. *onicframework.* [En línea] https://ionicframework.com/.
29. flask. *flask.* [En línea] http://flask.pocoo.org/.
30. wikipedia c lenguage. *wikipedia c lenguage.* [En línea] https://en.wikipedia.org/wiki/C_(programming_language).
31. Python, programming language. *Python, programming language.* [En línea] https://en.wikipedia.org/wiki/Python_(programming_language).
32. wikipedia pgsql. *wikipedia pgsql.* [En línea] https://en.wikipedia.org/wiki/PL/pgSQL.
33. wikipedia sistema d'informacio geografica. *wikipedia sistema d'informacio geografica.* [En línea] https://ca.wikipedia.org/wiki/Sistema_d%27informaci%C3%B3_geogr%C3%A0fica.
34. openstreetmap wiki. *openstreetmap wiki.* [En línea] https://wiki.openstreetmap.org/wiki/Main_Page.
35. OpenstreetMap wiki. *OpenstreetMap wiki.* [En línea] https://wiki.openstreetmap.org/wiki/Main_Page.

36. google developers, kml. *google developers, kml.* [En línea] https://developers.google.com/kml/.
37. wikipedia, shapefile. *wikipedia, shapefile.* [En línea] https://en.wikipedia.org/wiki/Shapefile.
38. wikipedia Gis file formats. *wikipedia Gis file formats.* [En línea] https://en.wikipedia.org/wiki/GIS_file_formats#Raster.
39. gdal, vrt. *gdal, vrt.* [En línea] https://gdal.org/drivers/raster/vrt.html.
40. postgis. *postgis.* [En línea] https://postgis.net/.
41. Openstreetmap wiki, Overpass API. *Openstreetmap wiki, Overpass API.* [En línea] https://wiki.openstreetmap.org/wiki/Overpass_API.
42. Openstreetmap wiki, JOSM. *Openstreetmap wiki, JOSM.* [En línea] https://wiki.openstreetmap.org/wiki/JOSM.
43. Openstreetmap wiki, Osm2pgsql. *Openstreetmap wiki, Osm2pgsql.* [En línea] https://wiki.openstreetmap.org/wiki/Osm2pgsql.
44. Qgis. *Qgis.* [En línea] https://qgis.org/ca/site/.
45. Institud geografic i geologic de catalunya. *Institud geografic i geologic de catalunya.* [En línea] https://www.icgc.cat/.
46. openstreetmap wiki, Osmosis. *openstreetmap wiki, Osmosis.* [En línea] https://wiki.openstreetmap.org/wiki/Osmosis.
47. Wikipedia, representational state transfer. *Wikipedia, representational state transfer.* [En línea] https://en.wikipedia.org/wiki/Representational_state_transfer.
48. Mehlhorn, Kurt y Sanders, Peter. Shortest Path. *Shortest Path.* [En línea] http://people.mpi-inf.mpg.de/~mehlhorn/ftp/Toolbox/ShortestPaths.pdf.
49. Geisberger, Robert. Contraction Hierarchies: Faster and Simpler Hierarchical Routing in Road Networks. [En línea] http://algo2.iti.kit.edu/documents/routeplanning/geisberger_dipl.pdf.
50. Postgis. Postgis, knn. [En línea] https://postgis.net/workshops/postgis-intro/knn.html.

10 Annexes

10.1 Annex 1- Tecnologies i llenguatges

10.1.1 Formats d'emmagatzematge d'informació geogràfica

10.1.1.1 OSM

OSM es el format dels fitxers de Open Street Map (un projecte col·laboratiu per crear un mapa lliure del món) Els fitxers OSM son fitxers en format XML de manera que estan etiquetats, estructurats i ordenats. Les etiquetes (tags) que més ens interessa son els 'node' que defineixen cada punt representat i els 'way' que representen els nexes entre nodes. Tant way com node poden tenir tags.

Per cada 'node' tenim informació, en concret ens interessa la seva posició geografica en forma dels atributs lat i lon (latitud i longitud), la altura está definida en osm amb el tag "ele" de 'elevation'. Aquest tag esta pensat per cims pero es pot fer servir per altres objectes. El seu valor ha de tenir com a unitat metres per sobre del nivell mitjà del mar com defineix el model geoide EGM96. En projectes anteriors s'ha afegit el tag 'elevation' que no es part dels tags reconeguts per evitar confusió ja que el tag 'ele' esta pensat per nodes on la altura és significativa. Aquesta arquitectura ens va molt bé per trobar quins camins realment estan connectats, ja que si ho estan tindran un node en comu.

Cada 'way' té definits amb els tags nd els nodes destí, ordenats per ordre de inserció, i també ens interessen els següents tags de tipus tag:

'highway': Defineix el tipus de carretera, camí o sender els senders que ens interes són(

path, motorway, trunk, primary, secondary, tertiary, unclassified, residential, motorway_link, trunk_link, primary_link, secondary_link, tertiary_link, living_street, service, pedestrian, track, road, footway, steps, path)

'Positive ramp': Definit per altres projectes, defineix en metres la altura que s'ha de pujar des del primer node definit nd fins el segon

'Negative Ramp': Definit per altres projectes, defineix en metres la altura que s'ha de baixar des del primer node definit nd fins el segon

‘DistanceFrom<node1>To<node2>’:Aquest tag on node1 i node 2 son els diferents identificadors dels nodes marca la distància en metres entre els dos nodes, el model osm te el tag ‘dist’ per definir distancia, pero esta reservat a aquells elements on dist tingui una importancia significativa. Fem servir aquest tag i no la distancia geométrica per coordenades, donat que el parc de montserrat té molt desnivell i la distància recorreguda canvia segons la altura.

‘oneway’: indica que el pas de vehicles nomes esta permes des de una direcció, si el valor es ‘yes’ la restriccio va en la direcció que s’ha dibuixat, en el cas contrari la direcció es en el sentit contrari

10.1.1.2 KML

KML es un format de fitxer per mostrar dades geogràfiques en eines com google earth. El format en que les dades es XML, es mantingut pel Open Geospatial Consortium, Inc. (OGC). En el cas de KML tots els tags son “case-sensitive”(les majúscules o minuscules son significatives) així que han de posar-se exactament com està especificat a la referencia de KML. La referencia explica quins tags existeixen, quins són opcionals i els tags han de seguir l’ordre d'aparició de la referencia.

En el cas de camins en kml et dona el tag LineString, que amb el subtag coordinates es defineix per on passa. Amb els tags extrude i tessellate pots definir si la linea ha de seguir el nivell del mar, el terreny o un valor absolut.

KML dons es un format molt adequat per representar formes sobre la superficie, pero com no te nodes no ens dóna informació sobre si dos punts realment es creuen o es troben sobre el pla en diferents altures a diferencia de OSM. Tot i que no és un format adequat per fer definir el graf com OSM. Els bombers tenen definit en aquest format la posició dels vehicles.

10.1.1.3 SHAPEFILE

El format Shapefile (desenvolupat per la companyia ESRI) és un format vectorial d'emmagatzematge digital on es guarda la localització dels elements geogràfics i els atributs associats a ells. Originalment es va crear per a la utilització amb el seu producte ArcView GIS, però actualment s'ha convertit en format estàndard de facto per a l'intercanvi d'informació geogràfica entre Sistemes d'Informació Geogràfica per la importància que els productes ESRI tenen al mercat SIG i per estar molt ben documentat.

És un format generat per diversos fitxers informàtics. El nombre mínim requerit és de tres i tenen les extensions següents:

.shp - és l'arxiu que emmagatzema les entitats geomètriques dels objectes.
.shx - és l'arxiu que emmagatzema l'índex de les entitats geomètriques.
.dbf - és la base de dades, en format dBase, on s'emmagatzema la informació dels atributs dels objectes.

A més d'aquestes extensions obligatòries n'hi ha d'altres on es pot definir informació complementària per millorar el funcionament en les operacions de consulta a la base de dades, informació sobre la projecció cartogràfica, o emmagatzematge de metadades.

Aquest és el format en que els bombers ens passen les seves dades cartogràfiques, de la informació que extraiem ens interessa el nom de la pista, que es correspon amb el tipus de pista seguint la tipologia de bombers, també la carpeta ens permet diferenciar entre les categories senders o pistes i vies.

Igual que kml te els punts representats en línies i no coneixem si dos línies que es troben estan realment a la mateixa altura. Per tant no ens acaba de servir per realitzar el graf

10.1.1.4 Fitxers raster

En la seva forma més simple, un ràster consisteix en una matriu de cel·les (o píxels) organitzat en files i columnes (o una graella) on cada cel·la conté un valor que representa informació, com ara la temperatura, o en el nostre cas altura. Els ràsters provenen de fotografies aèries digitals, imatges de satèl·lits, imatges digitals o fins i tot mapes escanejats.

Els ràsters ens permeten representar dades que canvien contínuament a través d'un paisatge. Proporcionen un mètode eficaç per emmagatzemar la continuïtat com a superfície. També proporcionen una representació de superfícies regularment espaiada. Els valors d'elevació mesurats des de la superfície terrestre són l'aplicació més comuna dels mapes de superfície.

10.1.2 LLenguatges

10.1.2.1 C

El llenguatge c, es un llenguatge compilat que tenint instruccions de alt nivell ens permet controlar els recursos de la màquina a baix nivell. El que ens permet optimitzar els accesos a memòria. Com en tota la part del graf els accesos a memoria juguen un paper important l'hem acabat implementant en c.

10.1.2.2 Python

Python es un llenguatge interpretat d'alt nivell,molt encarat a que el codi implementat sigui llegible. Tot i que es molt mes lent que c pel fet de ser un llenguatge interpretat, el fet que es senzill de implementar en ell, i la gran varietat de llibreries i facilitat de integrar-les l'han fet la millor opció per un projecte com aquest on s'ha de integrar moltes coses diferents. Entre les llibreries emprades, s'ha de destacar:

GDAL (Geospatial Data Abstraction Library) és una llibreria python de codi obert que ens permet llegir dades geogràfiques en formats raster i geoespacial de manera que ens permet convertir entre els diferents formats de coordenades(latitud i longitud cap a UTM) i conèixer l'altura de cert punt mitjançant la informació dels fitxers raster.

Ctypes ens permet cridar llibreries en c directament des del codi Python permetent-nos aprofitar la eficiència del codi c per les parts que ho requereixen sense perdre la llegibilitat i adaptabilitat de Python

10.1.2.3 Ionic

Ionic es un framework construït sobre angular, ens permet escriure en el llenguatge typescript (javascript tipificat) codi que ens permetrà obtenir una aplicació mòbil per Android o iOS o també una web per browser

10.1.2.4 postgresSQL

Postres es un projecte de programari lliure que implementa un sistema de gestió de bases de dades relacional. Amb l'extinció postgis ens aportarà una manera gestionar les dades referents a vies i punts

Postgis es una extensió de postgres, afegeix a postgres classes, funcions i tipus destinats a georeferenciar les dades

10.1.3 Eines de visualització i transformació de dades geogràfiques

10.1.3.1 API Overpass

La Api Overpass ens serveix per obtenir les dades seleccionades dins del model del mon de osm. Les dades es s'obtenen en format osm

10.1.3.2 JOSM

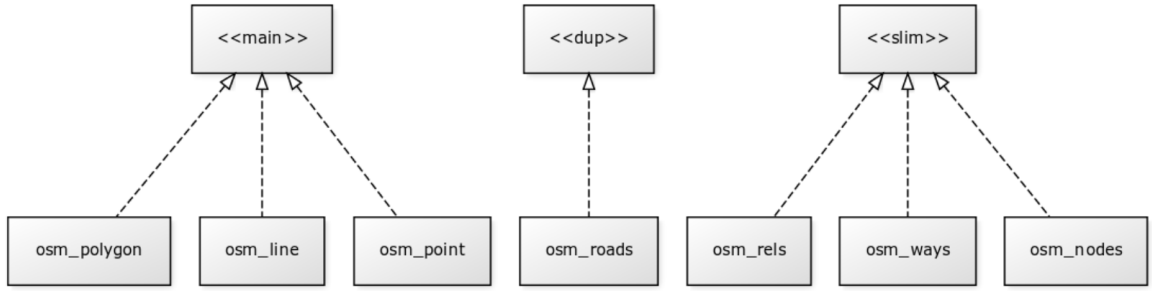
JOSM es una eina feta en java feta especialment per editar arxius osm. L'eina a part de permetre, crear i editar ways i nodes, te mecanismes de detecció i correcció d'errors. Ens permet detectar tant ways que es creuen sense tenir nodes comuns, com punts i linies sobreposats i seguidament ens dona l'opció d'arreglar-ho.Tambe ens permet sobreposar les dades a una imatge per satèl·lit de manera que podem sapiguer que el que estem fent es coerent amb la realitat

10.1.3.3 OSMOSIS

Osmosis ens permet filtrar les dades osm, es pot tant extreure dades per posició com per tipus. Això ens permet eliminar tota la informació que ni ens serà rellevant dels arxius OSM, i per tant no tenir que tractar una quantitat massa gran de dades

10.1.3.4 Osm2psql

Osm2psql ens permet carregar les dades de un fitxer osm a una base de dades postgis. Pero l'estructura per defecte que **Osm2psql** desa a la bd no ens interesa, ja que els camins i rutes els guarda en format geom (perdent la relacio dels nodes). Pero **Osm2psql** conta amb l'opció slim, que ens permet desar la relació de ways i nodes que tant ens interessa per a construir el graf en una estructura com la següent:



osm_polygon: conte les diferents formes, àreas representades dintre el osm importat
osm_line: conte les diferents línies, camins, ways, rutes representades en el osm importat
osm_point: conte els diferents punts representats en el osm importat

osm_roads: conte una simplificació de osm_line per evitar sobrecarregar el pintar en el cas de allunyar el zoom.
osm_rels:Conte les relacions(grup de tags) continguts dintre el osm importat amb la informació dels tags
osm_ways: Conte els ways i els nodes continguts dintre el osm importat amb la informació dels tags
osm_nodes: Conte la informació dels nodes continguts dintre el osm importat

10.1.3.5 QGIS

QGIS és una aplicació SIG professional que està integrada de ser programari lliure i de codi obert molt emprada en l'entorn cartografic, tambe s'emprea pel servei cartografic del cos de bombers.

Qgis ens permet interactuar amb les diferents capes tant de OSM com de KML i exportarles en el format que necessitem

10.1.3.6 Scripts i recursos del projecte anterior

Delprojecte anterior n'em extret els seguents scripts, i fitxers osm resultants d'executarlos:

DB2OSMpaths
Per passar aquest script és necesari tenir els camins a una BD Postgis. El script dona com a resultat un arxiu osm amb totes les rutes, extretes de les taules especificades al archiu **taulescat.txt** amb el tipus que també s'especifica al mateix archiu

Alçades
Un cop tenim el primer archiu osm el que farem es executar el script **dem3** que recorrera els punts del archiu osm i buscara la seva altura mitjançant un fitxer raster de la zona de Monserrat. Despres escriura un nou osm amb el tag "Elevation" indicant l'alçada del punt

OSM2NET
En el sistema anterior amb les alçades ja calculades podem calcular les distàncies i rampa acumulada de node a node, pero sent que la gran majoria de nodes nomes tenen dos arestes i que tots els nodes estan agrupats en ways que comparteixen totes les propietats, podem contar amb un graf mol simplificat si nomes calculem les distancies de les ways i ens quedem amb el primer hi ultim punt com origen i desti de la way. Per fer aixó totes les bifucarcions han de ser origen o desti i d'aixo s'encarrega aquest script

NET2GRAPH

Un cop ja tenim partits els ways ja podem quedarnos nomes amb els origens o destins, però aixo produira una perdua de informacio, aixi que al mateix temps aquest script calcula les distàncies rampa acumulada positiva i rampa acumulada negativa